A Multi-stage Stochastic Programming Approach for Network Capacity Expansion with Multiple Sources of Capacity

Majid Taghavi[a] and Kai Huang[a*]

November 9, 2015

[a] DeGroote School of Business, McMaster University, Hamilton, ON, Canada L8S4M4

Abstract

In networks, there are often more than one source of capacity. The capacities can be permanently or temporarily owned by the decision maker. Depending on the nature of sources, we identify the permanent capacity, spot market capacity and contract capacity. We use a scenario tree to model the uncertainty, and build a multi-stage stochastic integer program that can incorporate multiple sources and multiple types of capacities in a general network. We propose two solution methodologies for the problem. Firstly, we design an asymptotically convergent approximation algorithm. Secondly, we design a cutting plane algorithm based on Benders decomposition to find tight bounds for the problem. The numerical experiments show superb performance of the proposed algorithms compared with commercial software.

Keywords: capacity expansion, scenario tree, multi-stage stochastic integer programming, approximation algorithm, Benders decomposition, spot market, contract

*Corresponding author. E-mail: khuang@mcmaster.ca. Tel.: 1-905-525-9140. Fax: 1-905-525-8995.

1 Introduction

Capacity expansion is the problem of determining the optimal timing and amount of capacity acquisition, and the optimal capacity allocation, in order to address future demand growth. It is a long-term strategic decision which involves significant capital investment and uncertainties in forecasts. Capacity expansion has been widely studied in the literature since 1960s (Luss, 1984; Van Mieghem, 2003).

An important field of capacity expansion is the capacity expansion for networks, where the decision maker is interested in the planing of capacity expansion for network resources, e.g., arc flow capacity, node flow capacity, etc. The network capacity expansion problem has a wide range of applications in areas such as transportation (Magnanti and Wong, 1984; Ahuja et al., 1996; Liu et al., 2007; Marín and Jaramillo, 2008), telecommunication (Balakrishnan et al., 1991; Magnanti et al., 1995; Lee and Kang, 2000), service (Berman and Ganz, 1994), water distribution (Hsu et al., 2008), and manufacturing (Zhang et al., 2004).

In classical capacity expansion models, only permanent capacity is allowed. Permanent capacity refers to the capacity purchased and permanently owned by the decision maker. However, in practice, decision makers prefer to have a mixed procurement strategy that makes other sources of capacity available to them (Inderfurth and Kelle, 2011; Levin et al., 2012; Inderfurth et al., 2013). In this research, we introduce two sources of capacity for network resources besides the permanent capacity. These capacities could be beneficial, when they have a lower price, or when there is not enough permanent capacity to satisfy the demand. Spot market capacity refers to the capacity that can be obtained and used in the current period solely. It is temporary in nature and will be removed at the end of the current period. While spot market capacity brings a lot of flexibility to the decision maker, it also brings high risk of price uncertainty. Contract capacity refers to the capacity that is available in the current period, only if a contract has been signed for it in previous periods. The length of the contract are assumed to be fixed. Similar to the spot market capacity, the contract capacity has a temporary nature. However, our model allows the contracts to include multiple periods. It is noteworthy that when the contract only covers one period, spot market capacity and contract capacity are still different in nature. The former is obtained after the uncertainty in the current period is revealed, while the latter is signed before the uncertainty in the current period is revealed.

In this research, we use a scenario tree to model the uncertainty, and build a multi-stage stochastic integer program that can incorporate multiple sources[1] and multiple types of capacities in a general network. We study the case when there are only two sources of capacity (permanent capacity and spot market capacity), and the case of three sources of capacity (permanent capacity, spot market capacity and contract capacity). To solve our models, we propose an asymptotically convergent approximation algorithm, as well as a Benders decomposition based heuristic algorithm. Our numerical experiment results show that these algorithms are more efficient than commercial solver and could solve medium-scale and large-scale instances in reasonable amount of time.

The rest of the paper is organized as follows. In Section 2, we review the relevant literature. In Section 3, we study the stochastic network capacity expansion model with two sources of capacity. To solve it, we present an LP based approximation algorithm in Section 4. Next, in Section 5, we propose a Benders decomposition based heuristic algorithm to find tight bounds for the problems. In Section 6, we extend the stochastic network capacity expansion model to the case of three sources of capacity. We present the computational results for both algorithms in Section 7. Finally, we conclude the paper and discuss future research directions in Section 8.

2 Literature Review

The study of network capacity expansion problems started with the early works of Fulkerson (1959), Doulliez and Rao (1971), Christofides and Brooker (1974), Doulliez and Rao (1975), and Bansal and Jacobsen (1975). These early works are limited to the deterministic setting, where only permanent capacity is available.

In more recent studies with the same limitations, researchers studied deterministic network capacity expansion in the context of multi-commodity flow problem. Leung et al. (1990) studied capacity expansion of distribution centers in a route planning network. Their model considers node capacity expansion only. The original problem is decomposed into subprob-

[1]Note that the use of both source and resource could be confusing. Throughout this paper, source refers to the ownership type of capacity that is being purchased (permanent, spot market, and contract). Resource refers to the entity for which, we consider capacity expansion problem (e.g., node and arc).

lems, and solved by a Lagrangian relaxation approach. There are other works in the same context which considered arc capacity expansion only. Magnanti et al. (1995) studied telecommunication network capacity installation with two types of facilities, and generalized the results to multiple types of facilities. They proposed both Lagrangian relaxation, and a cutting plane approach with three sets of valid inequalities to solve the model. Bienstock and Günlük (1996) also proposed a mixed-integer programming formulation for a generic telecommunication network capacity installation with the objective of minimizing both capacity installation and flow costs. They studied the polyhedral structure of their model and provided extensions on the valid inequalities proposed in Magnanti et al. (1995). In another work, Bienstock et al. (1998) studied a special case of the generic model presented in Bienstock and Günlük (1996), where they considered a directed graph and tried to minimize the capacity installation cost only. They came up with different sets of valid inequalities and proposed two solution approaches and compared their results. Günlük (1999) considered additional capacity installation for arcs of a capacitated network and used an aggregate multicommodity flow formulation to model it. He developed a new branching technique and incorporated it into a branch-and-cut algorithm to solve the model. In a more relevant work, Ahuja et al. (1996) addressed arc capacity expansion in a transshipment network with budget constraints. Using a parametric network flow problem, they developed an efficient Simplex-based algorithm that can solve both linear and integer versions of the problem. Liu et al. (2007) proposed a railroad network model integrating yard location and yard capacity expansion decisions. Their model can capture both arc capacity (flow) and node capacity (car handling) and a greedy algorithm is presented to solve the integrated model. In this work, the additional capacity quantity is fixed and the decision is whether to add this quantity or not, while in our model, we decide on the quantity of capacity expansion.

With the advent of stochastic programming, researchers have considered stochastic network capacity expansion models. Sen et al. (1994) used a two-stage stochastic programming approach to model private line telecommunication networks. They considered arc capacity expansion and developed a stochastic decomposition algorithm to solve the problem. Pimentel et al. (2013) proposed a multi-stage stochastic mixed-integer programming model for a dynamic network design problem which integrates facility location, capacity expansion, and network design decisions. They considered capacity expansion of facilities (nodes) and used a scenario tree approach to

model the uncertainties of demands. They developed a Lagrangian heuristic procedure which can find feasible solutions with reasonably good bounds. Atamtürk and Zhang (2007), Mudchanatongsuk et al. (2007), Ordóñez and Zhao (2007), and Karoonsoontawong (2010) are examples of addressing arc capacity expansion problem with robust optimization approach. Another technique dealing with stochastic capacity expansion is Markov Decision Processes as illustrated in Bean et al. (1992), Bhatnagar et al. (1998), and Pratikakis (2010). However, to the best of our knowledge, there is no such work in stochastic network capacity expansion problems.

The are some works in the literature of capacity expansion in the presence of spot market and contract capacity. Atamtürk and Hochbaum (2001) studied subcontracting in deterministic multi-period capacity expansion problems. Oren et al. (2005) used game theory to model capacity expansion of electric power networks with spot market. Examples of applications in cellular manufacturing and flexible manufacturing systems can be found in Lee et al. (1997), Logendran and Puvanunt (1997), and Logendran and Ramakrishna (1997). A related but different research stream is capacity reservation with spot market, as appeared in Erkoc and Wu (2009) and Inderfurth et al. (2013).

In summary, to the best of our knowledge, there is no work on stochastic network capacity expansion that allows spot market and contract capacities simultaneously. Moreover, there are few studies that consider multiple resources simultaneously.

3 Model Development

In this section, we consider a multi-period, multi-resource stochastic capacity expansion model in the context of min-cost network flow problem. This model can be used in a wide range of network applications, where the decision maker has the choice to obtain the capacities from different sources. Here, we consider two sources of capacity: permanent and spot market.

In order to handle the uncertainties of the problem, we assume that our random parameters (costs and demands) evolve as discrete-time stochastic processes. We represent the stochastic processes as a scenario tree with T periods, as shown in Fig. 1, and assume that the number of possible states in each period is finite. For each node n in the scenario tree, $a(n)$ is the immediate ancestor node, and t_n is the period of node n. Let $\mathcal{T}$ be the whole scenario tree and N be the total number of nodes in the scenario tree.

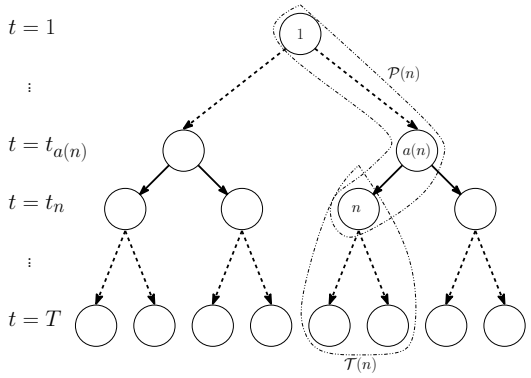

Figure 1: The scenario tree $\mathcal{T}$

$\mathcal{T}(n)$ denotes the subtree with root node n and $\bar{\mathcal{T}}(n) = \mathcal{T}(n) \setminus \{n\}$. $\mathcal{P}(n)$ denotes the unique path from node n to the root node of the scenario tree, and $\bar{\mathcal{P}}(n) = \mathcal{P}(n) \setminus \{n\}$.

Throughout this paper, we assume that the decision maker purchases the permanent capacity or signs the contract capacity at the beginning of a period, before the realization of the random demand, and spot market capacity is obtained after the the realization of the random demand in the same period. The purchased permanent or contract capacity will be available in the next period after it is purchased or signed. The spot market capacity will be available only in the period of purchase.

Consider an arbitrary network $G = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{A}$ is the set of arcs. We will consider capacity expansion of $|\mathcal{N}| + |\mathcal{A}|$ resources over T periods. In the model notations presented in Table 1, we use u and v to distinguish parameters and decision variables between arcs and nodes, respectively. In addition, note that index n refers to a node in the scenario tree $\mathcal{T}$. Furthermore, $\delta^+(i)$ is the set of arcs in $\mathcal{A}$ emanating from node i, and $\delta^-(i)$ is the set of arcs in $\mathcal{A}$ entering node i.

The stochastic network capacity expansion problem with two sources of capacity can be formulated as in (1)-(7). In this formulation, the objective function (1) minimizes the expected total flow cost and capacity acquisition costs related to nodes and arcs, over the scenario tree. Constraint (2) ensures that all demands are satisfied for all nodes in the scenario tree. Constraint (3) guarantees that the flow on each arc will not exceed the total permanent and spot market capacity acquired for that arc. Constraint (4) ensures

Table 1: Model notations for $\mathcal{SNCE}_{TWO}$ and $\mathcal{SNCE}_{THREE}$

Parameters:	
c^{up}_{nij}	Permanent capacity acquisition cost for arc $(i,j) \in \mathcal{A}$ at node $n \in \mathcal{T}$
c^{vp}_{ni}	Permanent capacity acquisition cost for node $i \in \mathcal{N}$ at node $n \in \mathcal{T}$
c^{us}_{nij}	Spot market capacity acquisition cost for arc $(i,j) \in \mathcal{A}$ at node $n \in \mathcal{T}$
c^{vs}_{ni}	Spot market capacity acquisition cost for node $i \in \mathcal{N}$ at node $n \in \mathcal{T}$
c^{uc}_{nij}	Contract capacity acquisition cost for arc $(i,j) \in \mathcal{A}$ at node $n \in \mathcal{T}$
c^{vc}_{ni}	Contract capacity acquisition cost for node $i \in \mathcal{N}$ at node $n \in \mathcal{T}$
c^{u}_{ij}	Flow cost on arc $(i,j) \in \mathcal{A}$
C^{u}_{ij}	Unit capacity of arc $(i,j) \in \mathcal{A}$
C^{v}_{i}	Unit capacity of node $i \in \mathcal{N}$
d_{ni}	Random demand (note: realized demand)for node $i \in \mathcal{N}$ at node $n \in \mathcal{T}$
p_n	Probability of node $n \in \mathcal{T}$
Decision variables:	
x_{nij}	Flow on arc $(i,j) \in \mathcal{A}$ at node $n \in \mathcal{T}$
u^{p}_{nij}	Permanent capacity acquisition for arc $(i,j) \in \mathcal{A}$ at node $n \in \mathcal{T}$
v^{p}_{ni}	Permanent capacity acquisition for node $i \in \mathcal{N}$ at node $n \in \mathcal{T}$
u^{s}_{nij}	Spot market capacity acquisition for arc $(i,j) \in \mathcal{A}$ at node $n \in \mathcal{T}$
v^{s}_{ni}	Spot market capacity acquisition for node $i \in \mathcal{N}$ at node $n \in \mathcal{T}$
u^{c}_{nij}	Contract capacity acquisition for arc $(i,j) \in \mathcal{A}$ at node $n \in \mathcal{T}$
v^{c}_{ni}	Contract capacity acquisition for node $i \in \mathcal{N}$ at node $n \in \mathcal{T}$

that the outgoing flow of each node cannot exceed the total permanent and spot market capacity acquired for that node. Non-negativity and integrality constraints are enforced through (5) to (7), respectively.

Model $\mathcal{SNCE}_{TWO}$:

$$\text{Min} \sum_{n\in\mathcal{T}} p_n \left[\sum_{(i,j)\in\mathcal{A}} (c^u_{ij} x_{nij} + c^{up}_{nij} u^p_{nij} + c^{us}_{nij} u^s_{nij}) + \sum_{i\in\mathcal{N}} (c^{vp}_{ni} v^p_{ni} + c^{vs}_{ni} v^s_{ni}) \right] \quad (1)$$

$$\text{s.t.} \sum_{(j,i)\in\delta_i^-} x_{nji} - \sum_{(i,j)\in\delta_i^+} x_{nij} = d_{ni} \qquad \forall i \in \mathcal{N}, \forall n \in \mathcal{T}, \quad (2)$$

$$x_{nij} \le C^u_{ij} \left(\sum_{m\in\bar{\mathcal{P}}(n)} u^p_{mij} + u^s_{nij} \right) \qquad \forall (i,j) \in \mathcal{A}, \forall n \in \mathcal{T}, \quad (3)$$

$$\sum_{(i,j)\in\delta_i^+} x_{nij} \le C^v_i \left(\sum_{m\in\bar{\mathcal{P}}(n)} v^p_{mi} + v^s_{ni} \right) \qquad \forall i \in \mathcal{N}, \forall n \in \mathcal{T}, \quad (4)$$

$$x_{nij} \in \mathbb{R}^+ \qquad \forall (i,j) \in \mathcal{A}, \forall n \in \mathcal{T}, \quad (5)$$

$$u^p_{nij},\ u^s_{nij} \in \mathbb{Z}^+ \qquad \forall (i,j) \in \mathcal{A}, \forall n \in \mathcal{T}, \quad (6)$$

$$v^p_{ni},\ v^s_{ni} \in \mathbb{Z}^+ \qquad \forall i \in \mathcal{N}, \forall n \in \mathcal{T}. \quad (7)$$

It is not hard to prove that $\mathcal{SNCE}_{TWO}$ is a difficult problem.

Theorem 1. *The stochastic program $\mathcal{SNCE}_{TWO}$ and its deterministic counterpart are NP-hard.*

Proof. See Appendix. □

Theorem 1 shows that the problem $\mathcal{SNCE}_{TWO}$ is intractable for large-scale instances. Therefore, in the following two sections, we study efficient approximation algorithms for large-scale instances.

4 The LP Based Approximation Algorithm

In this section, we develop an LP based approximation algorithm (LPA) for $\mathcal{SNCE}_{TWO}$ which returns near-optimal solutions, and can be proved to be asymptotically convergent to the optimal solution.

4.1 Subproblem: Single Resource Capacity Expansion

By considering the problem $\mathcal{SNCE}_{TWO}$, we noticed that it can be decomposed into capacity expansion problems for each node and each arc of the

network. We study these single resource stochastic capacity expansion problem, called $\mathcal{SCE}_{TWO}$, as a subproblem of $\mathcal{SNCE}_{TWO}$. As mentioned before, we assume that permanent capacity acquisition will be made at the beginning of each period, before the random data realization in that period; and spot market capacity acquisition will be made after the random data realization in the same period. This implies that the permanent capacity acquisition variable of node n will appear in constraints related to $\bar{\mathcal{T}}(n)$ and the spot market acquisition variable of node n will only appear in the constraint related to node n. The notations of $\mathcal{SCE}_{TWO}$ are presented in Table 2.

Table 2: Model notations for $\mathcal{SCE}_{TWO}$ and $\mathcal{SCE}_{THREE}$

Parameters:	
c_n^p	Permanent capacity acquisition cost at node $n \in \mathcal{T}$
c_n^s	Spot market capacity acquisition cost at node $n \in \mathcal{T}$
c_n^c	Contract capacity acquisition cost at node $n \in \mathcal{T}$
p_n	Probability of node $n \in \mathcal{T}$
d_n	Demand for node $n \in \mathcal{T}$
Decision variables:	
x_n	Permanent capacity acquisition at node $n \in \mathcal{T}$
z_n	Spot market capacity acquisition at node $n \in \mathcal{T}$
y_n	Contract capacity acquisition at node $n \in \mathcal{T}$

Without loss of generality, we assume that there is no previously acquired capacity. We also assume that d_n, c_n^s, and c_n^p are positive values for all $n \in \mathcal{T}$ and all costs are discounted to their present value. Given these assumptions, the stochastic single resource capacity expansion problem can be formulated as $\mathcal{SCE}_{TWO}$:

$$\begin{aligned} \mathcal{SCE}_{TWO}: \quad \text{Min} \quad & \sum_{n\in\mathcal{T}} p_n \left(c_n^p x_n + c_n^s z_n\right) & & \\ \text{s.t.} \quad & \sum_{m\in\bar{\mathcal{P}}(n)} x_m + z_n \geq d_n & \forall n \in \mathcal{T} & \qquad (8) \\ & x_n, z_n \in \mathbb{Z}^+ & \forall n \in \mathcal{T}, & \end{aligned}$$

where the objective minimizes the expected total acquisition cost of both permanent and spot market capacity, and the first set of constraints force

the total available capacity to be no less than the demand for each node. We can show that model $\mathcal{SCE}_{TWO}$ has the Total Unimodularity property (Wolsey and Nemhauser (1988)):

Theorem 2. *If the demands are integer-valued, the LP relaxation of $\mathcal{SCE}_{TWO}$ will yield integral optimal solutions.*

Proof. See Appendix. □

Theorem 2 shows that for integer demands, we can solve a linear relaxation of $\mathcal{SCE}_{TWO}$ instead of the original problem. We call the linear version $\mathcal{RSCE}_{TWO}$. For simplicity of exposition, we remove the node probabilities from its objective function. All algorithms in the following section are designed for $\mathcal{RSCE}_{TWO}$.

4.1.1 Polynomial-time Algorithms for the Subproblem

We develop two algorithm that can solve $\mathcal{RSCE}_{TWO}$ in polynomial time, one primal algorithm and one dual algorithm.

The primal algorithm is based on a shifting-up procedure. It compares the acquisition cost of permanent capacity in an ancestor node with the total acquisition cost of permanent and spot market capacities in descendant nodes; and if it is more economical, the capacities of descent nodes will be shifted to the ancestor node as permanent capacity. In this algorithm, we index the nodes in the scenario tree $\mathcal{T}$ based on the increasing order of their time periods and we call it the *primal indexing system*. The primal algorithm starts with an initial feasible solution where $x_n^* = 0$ and $z_n^* = d_n$ for all $n \in \mathcal{T}$. We also need the following definitions:

$$\begin{aligned}
\mathcal{A}(n) &= \left\{ m \in \bar{\mathcal{T}}(n) : x_m \text{ or } z_m > 0, \text{ and } x_k = 0, \forall k \in \bar{\mathcal{P}}(m) \setminus \mathcal{P}(n) \right\} \\
s_n &= \sum_{\substack{m \in \mathcal{A}(n) \\ x_m > 0}} c_m^p + \sum_{\substack{m \in \mathcal{A}(n) \\ z_m > 0}} c_m^s \\
\Delta_n &= \text{Min} \left\{ \min_{\substack{m \in \mathcal{A}(n) \\ x_m > 0}} x_m, \min_{\substack{m \in \mathcal{A}(n) \\ z_m > 0}} z_m \right\}
\end{aligned}$$

Given these definitions, the primal algorithm is presented in Algorithm 1.

Algorithm 1 - The Primal Algorithm for $\mathcal{RSCE}_{TWO}$

set $x_n^* = 0$ and $z_n^* = d_n, \quad \forall n \in \mathcal{T}$
set $k = \text{Max}\{n \in \mathcal{T} : \bar{\mathcal{T}}(n) \neq \emptyset\}$
while $k \geq 1$ **do**
    compute $\mathcal{A}(k)$, s_k and Δ_k
    while $c_k^p \leq s_k$ **do**
        $$x_m^* = \begin{cases} x_m^* + \Delta_k & \text{if } m = k, \\ x_m^* - \Delta_k & \text{if } m \in \mathcal{A}(k) \text{ and } x_m^* > 0. \end{cases}$$
        $z_m^* = z_m^* - \Delta_k \qquad \text{if } m \in \mathcal{A}(k) \text{ and } z_m^* > 0$
        update $\mathcal{A}(k)$, s_k and Δ_k
    end while
    $k = k - 1$
end while
return $x^* = (x_1^*, \cdots, x_N^*)$ and $z^* = (z_1^*, \cdots, z_N^*)$

The dual algorithm is designed to solve the dual of the single resource capacity expansion problem. If π_n is the dual variable, the dual problem of $\mathcal{RSCE}_{TWO}$ is:

$$\begin{array}{lrcll} \text{Max} & \displaystyle\sum_{n \in \mathcal{T}} d_n \pi_n & & & \\ \text{s.t.} & \displaystyle\sum_{m \in \bar{\mathcal{T}}(n)} \pi_m & \leq & c_n^p & \forall n \in \mathcal{T}, \\ & \pi_n & \leq & c_n^s & \forall n \in \mathcal{T}, \\ & \pi_n & \geq & 0 & \forall n \in \mathcal{T}. \end{array} \tag{9}$$

It can be proved (see Appendix) that any feasible solution of the dual problem will satisfy $\pi_n \leq \text{Min}\left\{c_n^s, \text{Min}_{m \in \bar{\mathcal{P}}(n)}\left\{c_m^p - \sum_{k \in \bar{\mathcal{T}}(m) \setminus \{n\}} \pi_k\right\}\right\}$, and the equality holds at optimality. This observation motivates us to design a greedy algorithm to solve the dual problem. To implement it, we assume that all demands are different and we index the nodes in decreasing order of their demand, i.e., $d_1 > d_2 > \cdots > d_N$. This indexing scheme is called the *dual indexing system*. The dual algorithm is given in Algorithm 2.

Algorithm 2 - The Dual Algorithm for the Dual of $\mathcal{RSCE}_{TWO}$

1: set $\pi_n^* = 0$ and $c_n^0 = c_n^p, \quad \forall n \in \mathcal{T}$

2: **for** $k = 1, 2, \cdots, N$ **do**

3: $$\pi_k^* = \begin{cases} \underset{m \in \bar{\mathcal{P}}(k)}{\text{Min}} \left\{ c_m^{k-1} \right\} & \text{if } \text{Min}_{m \in \bar{\mathcal{P}}(k)} \left\{ c_m^{k-1} \right\} \leq c_k^s, \\ c_k^s & \text{otherwise.} \end{cases}$$

4: $$c_n^k = \begin{cases} c_n^{k-1} - \pi_k^* & \text{if } n \in \bar{\mathcal{P}}(k), \\ c_n^{k-1} & \text{otherwise.} \end{cases}$$

5: **end for**

6: return $\pi^* = (\pi_1^*, \pi_2^*, \cdots, \pi_N^*)$

The next theorem shows that the solutions returned by the primal and the dual algorithms are optimal.

Theorem 3. *The solution* $(x^*, z^*) = (x_1^*, x_2^*, \cdots, x_N^*, z_1^*, z_2^*, \cdots, z_N^*)$ *returned by the primal algorithm and the solution* $\pi^* = (\pi_1^*, \pi_2^*, \cdots, \pi_N^*)$ *returned by the dual algorithm are optimal.*

Proof. For the proof, we state Lemma 1-6 and prove them. See Appendix. □

4.1.2 Complexities of the Polynomial-time Algorithms

Suppose that the scenario tree is a complete tree with T levels (i.e., periods) and B branches for every non-leaf node. Then the number of nodes in the scenario tree is $N = \sum_{t=0}^{T-1} B^t = \frac{B^T - 1}{B-1}$, which implies that $T \sim \mathcal{O}(\log N)$.

Theorem 4. *The complexity of the primal algorithm is* $\mathcal{O}(N^2)$ *and the complexity of the dual algorithm is* $\mathcal{O}(N \log N \log(\log N))$.

Proof. See Appendix. □

In the next subsection, we present the LPA algorithm which incorporates the solution techniques for the single resource capacity expansion problem.

4.2 The LPA Algorithm

As we presented in Section 4.1, $\mathcal{SNCE}_{TWO}$ can be decomposed into smaller problems, which is the key to our LPA algorithm. Ahmed and Sahinidis

(2003) and Huang and Ahmed (2009) also explored decomposition structures to solve multi-stage stochastic capacity expansion problems. However, our method is different in that it includes multiple sources of capacity, which makes the analysis relevant to costs of multiple sources and different from previous works. For simplicity of exposition, let $\mathcal{X} = (\mathbf{x}_n)_{n\in\mathcal{T}}$ in which $\mathbf{x}_n = (x_{nij})_{(i,j)\in\mathcal{A}}$ and $\mathcal{Y} = (\mathbf{y}_n)_{n\in\mathcal{T}} = (\mathbf{u}^{\mathbf{p}}_n, \mathbf{v}^{\mathbf{p}}_n, \mathbf{u}^{\mathbf{s}}_n, \mathbf{v}^{\mathbf{s}}_n)_{n\in\mathcal{T}}$ where $\mathbf{u}^{\mathbf{p}}_n = (u^p_{nij})_{(i,j)\in\mathcal{A}}$, $\mathbf{v}^{\mathbf{p}}_n = (v^p_{ni})_{i\in\mathcal{N}}$, $\mathbf{u}^{\mathbf{s}}_n = (u^s_{nij})_{(i,j)\in\mathcal{A}}$, and $\mathbf{v}^{\mathbf{s}}_n = (v^s_{ni})_{i\in\mathcal{N}}$.

We start with an illustration of the decomposition structure. Note that the problem $\mathcal{SNCE}_{TWO}$ is equivalent to:

$$\begin{aligned}
\text{Min} \quad & \sum_{n\in\mathcal{T}} p_n \sum_{(i,j)\in\mathcal{A}} c^u_{ij} x_{nij} + \sum_{i\in\mathcal{N}} Q^1_i(\mathcal{X}) + \sum_{(i,j)\in\mathcal{A}} Q^2_{(i,j)}(\mathcal{X}) \\
\text{s.t.} \quad & \sum_{(j,i)\in\delta^-_i} x_{nji} - \sum_{(i,j)\in\delta^+_i} x_{nij} = d_{ni} && \forall i \in \mathcal{N}, \forall n \in \mathcal{T}, \\
& x_{nij} \in \mathbb{R}^+ && \forall (i,j) \in \mathcal{A}, \forall n \in \mathcal{T},
\end{aligned} \tag{10}$$

where

$$\begin{aligned}
Q^1_i(\mathcal{X}) = \text{Min} \quad & \sum_{n\in\mathcal{T}} p_n \left(c^{vp}_{ni} v^p_{ni} + c^{vs}_{ni} v^s_{ni}\right) \\
\text{s.t.} \quad & \sum_{m\in\bar{\mathcal{P}}(n)} v^p_{mi} + v^s_{ni} \geq \frac{\sum_{(i,j)\in\delta^+_i} x_{nij}}{C^v_i} && \forall n \in \mathcal{T}, \\
& v^p_{ni}, v^s_{ni} \in \mathbb{Z}^+ && \forall n \in \mathcal{T},
\end{aligned} \tag{11}$$

and

$$\begin{aligned}
Q^2_{(i,j)}(\mathcal{X}) = \text{Min} \quad & \sum_{n\in\mathcal{T}} p_n \left(c^{up}_{nij} u^p_{nij} + c^{us}_{nij} u^s_{nij}\right) \\
\text{s.t.} \quad & \sum_{m\in\bar{\mathcal{P}}(n)} u^p_{mij} + u^s_{nij} \geq \frac{x_{nij}}{C^u_{ij}} && \forall n \in \mathcal{T}, \\
& u^p_{nij}, u^s_{nij} \in \mathbb{Z}^+ && \forall n \in \mathcal{T}.
\end{aligned} \tag{12}$$

The problem (10) involves the capacity allocation decisions, while (11) involves the capacity acquisition for nodes of the network, and (12) involves the capacity acquisition for arcs of the network. Note that for a fixed capacity allocation decision $\mathcal{X} = (\mathbf{x}_n)_{n\in\mathcal{T}}$, (11) and (12) can be seen as the single resource capacity expansion subproblems $\mathcal{SCE}_{TWO}$. Both models can be solved using the polynomial-time algorithms developed in Section 4.1.

Having identified the sub-structure of the problem $\mathcal{SNCE}_{TWO}$, we propose the LPA algorithm outlined in Algorithm 3. In Step 1, the LP relaxation of $\mathcal{SNCE}_{TWO}$ is solved to optimality. This problem is a multi-stage stochastic linear program. Step 2 requires the solution of $|\mathcal{N}| + |\mathcal{A}|$ single resource stochastic capacity expansion subproblems, which are multi-stage stochastic integer programs. It is noteworthy that if we replace the right-hand side in constraints of (11) by $\left\lceil \frac{\sum_{(i,j)\in\delta_i^+} x_{nij}}{C_i^v} \right\rceil$, and the right-hand side in constraints of (12) by $\left\lceil \frac{x_{nij}}{C_{ij}^u} \right\rceil$, then the programs can be solved using the primal and dual algorithms developed in Section 4.1. Finally, Step 3 requires the solution of N (for all nodes in the tree) independent linear capacity allocation problems, which are min-cost flow problems.

We can prove that the LPA algorithm has an asymptotic convergence property. First, we present an upper bound for the gap between the optimal solution and the solution returned by the LPA algorithm:

Theorem 5. *For a solution $(\mathcal{X}, \mathcal{Y})$, let $f(\mathcal{X}, \mathcal{Y})$ be the objective function. If $(\mathcal{X}^*, \mathcal{Y}^*)$ is the optimal solution of $\mathcal{SNCE}_{TWO}$ and $(\mathcal{X}^A, \mathcal{Y}^A)$ is the solution returned by the LPA algorithm, then,*

$$f(\mathcal{X}^A, \mathcal{Y}^A) - f(\mathcal{X}^*, \mathcal{Y}^*) \leq \sum_{(i,j)\in\mathcal{A}} \left(c_{1ij}^{up} + c_{1ij}^{us}\right) + \sum_{i\in\mathcal{N}} \left(c_{1i}^{vp} + c_{1i}^{vs}\right).$$

Proof. See Appendix. □

Theorem 5 shows that the optimality gap of the solution returned by Algorithm 3 is bounded by the sum of the permanent and spot market capacity acquisition costs of arcs and nodes of the network at the root node of the scenario tree, and is independent of other model parameters such as the number of time periods, the number of branches in the scenario tree, etc. This property leads to the following asymptotic convergence result:

Theorem 6. *Assume that:*

(i) There exists $\epsilon_1 > 0$ such that for all $n \in \mathcal{T}$, there is at least one node in the network whose demand is at least ϵ_1;

(ii) There exists $\epsilon_2 > 0$ such that for all $n \in \mathcal{T}$, and for any arc $(i', j') \in A$ with $\sum_{(j',i')\in\delta_{i'}^-} x_{nj'i'} - \sum_{(i',j')\in\delta_{i'}^+} x_{ni'j'} < 0$, the total capacity allocation cost $c_{i'j'}^u + c_{i'}^v \geq \epsilon_2 d_{ni'}$;

Algorithm 3 - The LPA Algorithm for $\mathcal{SNCE}_{TWO}$

1: Solve the LP relaxation of $\mathcal{SNCE}_{TWO}$ and Let $(\mathcal{X}^{LP}, \mathcal{Y}^{LP}) := (\mathbf{x}_n^{LP}, \mathbf{y}_n^{LP})_{n\in\mathcal{T}}$. If $\mathbf{y}_n^{LP}$ is integral for all n, stop and return $(\mathcal{X}^{LP}, \mathcal{Y}^{LP})$.

2: Fix the capacity allocation decisions $\mathcal{X}^{LP}$. For each resource, solve the independent single resource capacity expansion problem, i.e., for each node $i \in \mathcal{N}$, solve (11) with $\left\lceil \frac{\sum_{(i,j)\in\delta_i^+} x_{nij}}{C_i^v} \right\rceil$ as the right-hand side, and for each arc $(i,j) \in \mathcal{A}$, solve (12) with $\left\lceil \frac{x_{nij}}{C_{ij}^u} \right\rceil$ as the right-hand side. Let $\mathcal{Y}^A = (\mathbf{y}_n^A)_{n\in\mathcal{T}} = (\mathbf{u}_n^{\mathbf{p}A}, \mathbf{v}_n^{\mathbf{p}A}, \mathbf{u}_n^{\mathbf{s}A}, \mathbf{v}_n^{\mathbf{s}A})_{n\in\mathcal{T}}$ denote the corresponding optimal solution.

3: Fix the capacity allocation decisions $\mathcal{Y}^A$. For each $n \in \mathcal{T}$, solve the independent network flow problem (13) and let $\mathbf{x}_n^A$ denote the corresponding optimal solution, and $\mathcal{X}^A = (\mathbf{x}_n^A)_{n\in\mathcal{T}}$.

$$
\begin{aligned}
\text{Min} \quad & \sum_{(i,j)\in\mathcal{A}} c_{ij}^u x_{nij} + \sum_{i\in\mathcal{N}} \sum_{(i,j)\in\delta^+(i)} x_{nij} & & \\
\text{s.t.} \quad & \sum_{(j,i)\in\delta_i^-} x_{nji} - \sum_{(i,j)\in\delta_i^+} x_{nij} = d_{ni} & & \forall i \in \mathcal{N}, \\
& x_{nij} \le C_{ij}^u \left(\sum_{m\in\bar{\mathcal{P}}(n)} u_{mij}^{pA} + u_{nij}^{sA} \right) & & \forall (i,j) \in \mathcal{A}, \\
& \sum_{(i,j)\in\delta_i^+} x_{nij} \le C_i^v \left(\sum_{m\in\bar{\mathcal{P}}(n)} v_{mi}^{pA} + v_{ni}^{sA} \right) & & \forall i \in \mathcal{N}, \\
& x_{nij} \in \mathbb{R}^+ & & \forall (i,j) \in \mathcal{A}.
\end{aligned}
\tag{13}
$$

4: Return $(\mathcal{X}^A, \mathcal{Y}^A) := (\mathbf{x}_n^A, \mathbf{y}_n^A)_{n\in\mathcal{T}}$.

then

$$\lim_{T\to\infty} \frac{f(\mathcal{X}^A, \mathcal{Y}^A) - f(\mathcal{X}^*, \mathcal{Y}^*)}{f(\mathcal{X}^*, \mathcal{Y}^*)} = 0$$

for $\mathcal{SNCE}_{TWO}$.

Proof. According to Theorem 5, we have:

$$\frac{f(\mathcal{X}^A, \mathcal{Y}^A) - f(\mathcal{X}^*, \mathcal{Y}^*)}{f(\mathcal{X}^*, \mathcal{Y}^*)} \leq \frac{\sum_{(i,j)\in\mathcal{A}} \left(c^{up}_{1ij} + c^{us}_{1ij}\right) + \sum_{i\in\mathcal{N}} \left(c^{vp}_{1i} + c^{vs}_{1i}\right)}{f(\mathcal{X}^*, \mathcal{Y}^*)}.$$

Therefore, we only need to show that $\lim_{T\to\infty} f(\mathcal{X}^*, \mathcal{Y}^*) = \infty$. For a fixed T, consider the dual of the linear relaxation of $\mathcal{SNCE}_{TWO}$ with dual variables π^d_{ni}, π^u_{nij}, and π^v_{ni} for (2), (3), and (4), respectively:

$$\begin{aligned}
\text{Max} \quad & \sum_{n\in\mathcal{T}} \pi^d_{ni} d_{ni} \\
\text{s.t.} \quad & -\pi^d_{ni} + \pi^d_{nj} - \pi^u_{nij} - \pi^v_{ni} \leq p_n c^u_{ij} && \forall (i,j) \in \mathcal{A}, \forall n \in \mathcal{T}, \\
& C^u_{ij} \sum_{m\in\bar{\mathcal{T}}(n)} \pi^u_{nij} \leq p_n c^{up}_{nij} && \forall (i,j) \in \mathcal{A}, \forall n \in \mathcal{T}, \\
& C^v_i \sum_{m\in\bar{\mathcal{T}}(n)} \pi^v_{ni} \leq p_n c^{vp}_{ni} && \forall i \in \mathcal{N}, \forall n \in \mathcal{T}, \\
& C^u_{ij} \pi^u_{nij} \leq p_n c^{us}_{nij} && \forall (i,j) \in \mathcal{A}, \forall n \in \mathcal{T}, \\
& C^v_i \pi^v_{ni} \leq p_n c^{vs}_{ni} && \forall i \in \mathcal{N}, \forall n \in \mathcal{T}, \\
& \pi^d_{ni},\ \pi^v_{ni} \in \mathbb{R}^+ && \forall i \in \mathcal{N}, \forall n \in \mathcal{T}, \\
& \pi^u_{nij} \in \mathbb{R}^+ && \forall (i,j) \in \mathcal{A}, \forall n \in \mathcal{T}.
\end{aligned}$$

Let the vector form of the dual variables be π^d_n, π^u_n, and π^v_n. For any dual feasible solution $(\pi^d_n, \pi^u_n, \pi^v_n)$, let $g(\pi^d_n, \pi^u_n, \pi^v_n) = \sum_{n\in\mathcal{T}} \pi^d_{ni} d_{ni}$ denote the objective value of the above dual problem. Clearly, weak duality ensures that $f(\mathcal{X}^{LP}, \mathcal{Y}^{LP}) \geq g(\pi^d_n, \pi^u_n, \pi^v_n)$. Now, we try to find a dual feasible solution $(\bar{\pi}^d_n, \bar{\pi}^u_n, \bar{\pi}^v_n)$ such that $g(\bar{\pi}^d_n, \bar{\pi}^u_n, \bar{\pi}^v_n) \geq T\epsilon$ for some $\epsilon > 0$. We set $\bar{\pi}^u_n = \bar{\pi}^v_n = 0$, for all $n \in \mathcal{T}$. Assumption (i) requires that there is at least one node $i' \in \mathcal{N}$ such that $-d_{ni'} \geq \epsilon_1$. So, constraint (2) ensures that there is at least one node i' such that $\sum_{(j',i')\in\delta^-_{i'}} x_{nj'i'} - \sum_{(i',j')\in\delta^+_{i'}} x_{ni'j'} < 0$; and we can find at least one emanating arc (i', j') of node i'. Now, if we let $\bar{\pi}^d_{ni'} = -p_n(c^u_{i'j'} + c^v_{i'})$ and $\bar{\pi}^d_{ni} = 0$ for all $i \in \mathcal{N}\backslash\{j'\}$, then $(\bar{\pi}^d_n, \bar{\pi}^u_n, \bar{\pi}^v_n)$ is a dual feasible solution. Also, assumption (ii) requires that $-\bar{\pi}^d_{ni'} \geq p_n\epsilon_2$. Therefore, $g(\bar{\pi}^d_n, \bar{\pi}^u_n, \bar{\pi}^v_n) = \sum_{n\in\mathcal{T}} \bar{\pi}^d_{ni} d_{ni} = \sum_{n\in\mathcal{T}} \bar{\pi}^d_{ni'} d_{ni'} \geq \sum_{n\in\mathcal{T}} p_n\epsilon_1\epsilon_2$. Since the summation of probabilities of each stage in the scenario tree is 1 and we have T stages, $\sum_{n\in\mathcal{T}} p_n = T$, which means that $g(\bar{\pi}^d_n, \bar{\pi}^u_n, \bar{\pi}^v_n) \geq T\epsilon_1\epsilon_2$. Using weak duality, we can conclude that $f(\mathcal{X}^{LP}, \mathcal{Y}^{LP}) \geq T\epsilon_1\epsilon_2$. Therefore, $\lim_{T\to\infty} f(\mathcal{X}^{LP}, \mathcal{Y}^{LP}) = \infty$. □

We have shown that the LPA algorithm will converge to the optimal solution when the number of periods increases.

5 The Benders Decomposition Based Heuristic Algorithm

Although the LPA algorithm presented in Section 4 is very efficient in solving $\mathcal{SNCE}_{TWO}$, it requires the number of scenario tree periods to be large so that it converges to the optimal solution. This may lead to having huge scenario trees, especially when the number of branches is also large. To overcome this difficulty, in this section, we consider using Benders decomposition techniques.

The classical Benders decomposition methods used for multi-stage stochastic programming problems decompose the problem by scenarios. For our model $\mathcal{SNCE}_{TWO}$, the master problem will contain all the capacity acquisition variables (u^p_{nij}, u^s_{nij}, v^p_{ni}, and v^s_{ni}) and by fixing those integer variables, we can obtain N subproblems (each corresponding to a node in the scenario tree) containing flow variables (x_{nij}). After solving each subproblem, if it is unbounded, a feasibility cut will be added to the master problem; if all subproblems are optimal, then optimality cuts will be added to the master problem.

Our preliminary experimental results showed that this algorithm converges very slowly compared to CPLEX MIP solver. To improve the efficiency of our Benders algorithm, we implemented some improvement techniques. First, we introduced three set of valid inequalities for the master problem. For each $n \in \mathcal{T}$ and for each origin node i with $d_{ni} < 0$, $-d_{ni} \leq C^v_i \left(\sum_{m \in \bar{\mathcal{P}}(n)} v^p_{mi} + v^s_{ni} \right)$ is a valid inequality, and for each origin node i with $d_{ni} < 0$, $-d_{ni} \leq \sum_j C^u_{ij} \left(\sum_{m \in \bar{\mathcal{P}}(n)} u^p_{mij} + u^s_{nij} \right)$ is a valid inequality, and for each destination node i with $d_{ni} > 0$, $d_{ni} \leq \sum_j C^u_{ji} \left(\sum_{m \in \bar{\mathcal{P}}(n)} u^p_{mji} + u^s_{nji} \right)$ is a valid inequality. Second, we incorporated some well-known improvement techniques to Benders decomposition algorithm such as solving a linear relaxation of the master problem (McDaniel and Devine, 1977), adding Pareto-optimal cuts (Magnanti and Wong, 1981), and using the maximum feasible subsystem (Saharidis and Ierapetritou, 2010). Although the Benders decomposition solution time has been improved using these techniques, it still could

not outperform the CPLEX MIP solver. Motivated by this observation, we design a Benders decomposition based heuristic (BDH) algorithm to solve our models.

In the BDH algorithm, we decompose the problem by resources corresponding to nodes and arcs of the network. By doing this, the master problem will be a linear program containing flow variables and all $|\mathcal{N}| + |\mathcal{A}|$ subproblems will be integer programs including capacity variables. Since these subproblems have Totally Unimodular property (see Section 4), we can solve the linear relaxation of them provided that the right-hand side of their constraints are integer-valued.

Based on this observation, the initial master problem for $\mathcal{SNCE}_{TWO}$ can be modeled as follows:

$$\begin{aligned}
\text{Min} \quad & \sum_{n\in\mathcal{T}} p_n \sum_{(i,j)\in\mathcal{A}} c^u_{ij} x_{nij} + \sum_{(i,j)\in\mathcal{A}} \eta_{ij} + \sum_{i\in\mathcal{N}} \theta_i & & \\
\text{s.t.} \quad & \sum_{(j,i)\in\delta^-_i} x_{nji} - \sum_{(i,j)\in\delta^+_i} x_{nij} = d_{ni} & \forall i \in \mathcal{N}, \forall n \in \mathcal{T}, & \quad (14)\\
& \theta_i \in \mathbb{R}^+ & \forall i \in \mathcal{N}, \forall n \in \mathcal{T}, & \\
& x_{nij},\ \eta_{ij} \in \mathbb{R}^+ & \forall (i,j) \in \mathcal{A}, \forall n \in \mathcal{T}, &
\end{aligned}$$

where η and θ are auxiliary variables corresponding to arcs and nodes, respectively. We define subproblems for each resource of the network. For each arc (i,j) and for a fixed value of $\mathcal{X}$, the subproblem is $\mathcal{SP}_{(i,j)}$:

$$\begin{aligned}
\mathcal{SP}_{(i,j)}(\mathcal{X}) = \text{Min} \quad & \sum_{n\in\mathcal{T}} p_n (c^{up}_{nij} u^p_{nij} + c^{us}_{nij} u^s_{nij}) & & \\
\text{s.t.} \quad & \sum_{m\in\bar{\mathcal{P}}(n)} u^p_{mij} + u^s_{nij} \geq \left\lceil \frac{x_{nij}}{C^u_{ij}} \right\rceil & \forall n \in \mathcal{T}, & \\
& u^p_{nij},\ u^s_{nij} \in \mathbb{R}^+ & \forall n \in \mathcal{T}, &
\end{aligned}$$

and for each node i and for a fixed value of $\mathcal{X}$, the subproblem is $\mathcal{SP}_i$:

$$\begin{aligned}
\mathcal{SP}_i(\mathcal{X}) = \text{Min} \quad & \sum_{n\in\mathcal{T}} p_n (c^{vp}_{ni} v^p_{ni} + c^{vs}_{ni} v^s_{ni}) & & \\
\text{s.t.} \quad & \sum_{m\in\bar{\mathcal{P}}(n)} v^p_{mi} + v^s_{ni} \geq \left\lceil \frac{\sum_{(i,j)\in\delta^+_i} x_{nij}}{C^v_i} \right\rceil & \forall n \in \mathcal{T}, & \\
& v^p_{ni},\ v^s_{ni} \in \mathbb{R}^+ & \forall n \in \mathcal{T}. &
\end{aligned}$$

Note that for both arc and node subproblems, the right-hand side is integer-valued, so that we can relax the integrality constraints. Furthermore, for any $\mathcal{X}$ returned by the master problem, these subproblems are feasible. A major advantage of this method is that these subproblems are always feasible and solving them always result in optimality cuts for the master problem. These optimality cuts can be derived from solving the dual of these problems. For brevity of exposition, we don't present the formulations of the dual subproblems. Suppose π_{nij} and π_{ni} are dual variables for arc and node dual subproblems, respectively. Solving arc dual subproblem will yield optimality cut of the form

$$\sum_{n\in\mathcal{T}} \left\lceil \frac{x_{nij}}{C^u_{ij}} \right\rceil \pi^*_{nij} \leq \eta_{ij}, \tag{15}$$

and solving node dual subproblem will yield optimality cut of the form

$$\sum_{n\in\mathcal{T}} \left\lceil \frac{\sum_{(i,j)\in\delta^+_i} x_{nij}}{C^v_i} \right\rceil \pi^*_{ni} \leq \theta_i, \tag{16}$$

where π^*_{nij} and π^*_{ni} are dual optimal solutions. Unfortunately, both (15) and (16) are non-linear cuts and cannot be used in Benders decomposition procedure. However, by proving the following theorem, we can modify them to find bounds using a Benders decomposition procedure.

Theorem 7. *If we use the cuts*

$$\sum_{n\in\mathcal{T}} \frac{x_{nij}}{C^u_{ij}} \pi^*_{nij} \leq \eta_{ij} \tag{17}$$ *and* $$\sum_{n\in\mathcal{T}} \frac{\sum_{(i,j)\in\delta^+_i} x_{nij}}{C^v_i} \pi^*_{ni} \leq \theta_i, \tag{18}$$

then we will obtain a lower bound of the original master problem, and if we use the cuts

$$\sum_{n\in\mathcal{T}} \left(\frac{x_{nij}}{C^u_{ij}} + 1 \right) \pi^*_{nij} \leq \eta_{ij} \tag{19}$$ *and* $$\sum_{n\in\mathcal{T}} \left(\frac{\sum_{(i,j)\in\delta^+_i} x_{nij}}{C^v_i} + 1 \right) \pi^*_{ni} \leq \theta_i, \tag{20}$$

then we will obtain an upper bound of the original master problem.

Proof. Since $\pi^*_{nij} \geq 0$ and $\pi^*_{ni} \geq 0$, for all $n \in \mathcal{T}$, we have $\left\lceil \frac{x_{nij}}{C^u_{ij}} \right\rceil \geq \frac{x_{nij}}{C^u_{ij}}$ and $\left\lceil \frac{\sum_{(i,j)\in\delta^+_i} x_{nij}}{C^v_i} \right\rceil \geq \frac{\sum_{(i,j)\in\delta^+_i} x_{nij}}{C^v_i}$, which means cut (15) dominates cut (17) and cut (16) dominates cut (18). Similarly, we can show that cuts (19) and (20) dominate cuts (15) and (16), respectively. □

Since the cuts that result in finding lower bound cannot be used in the same master problem with the ones that result in finding upper bounds, we need to keep two master problems. Corresponding to each set of cuts, we define $\mathcal{MASTER}_{\mathcal{LB}}$ and $\mathcal{MASTER}_{\mathcal{UB}}$. Initially, both master problems are of the form (14). However, at each iteration, cuts of (17) and (18) will be added to $\mathcal{MASTER}_{\mathcal{LB}}$ and cuts of (19) and (20) will be added to $\mathcal{MASTER}_{\mathcal{UB}}$.

Now, let $Z^*_{\mathcal{LB}}$ be the objective function value of the $\mathcal{MASTER}_{\mathcal{LB}}$ in each iteration. Also, suppose Z^L_{ij} and Z^L_i are the objective function values of arc (i, j) and node i subproblems solved using the incumbent values returned by $\mathcal{MASTER}_{\mathcal{LB}}$, respectively. Similarly, Z^U_{ij} and Z^U_i are the objective function values of arc (i, j) and node i subproblems solved using the incumbent values returned by $\mathcal{MASTER}_{\mathcal{UB}}$, respectively. Given these explanations, we present the BDH algorithm formally in Algorithm 4.

It is important to note that the structure of the node and arc subproblems guarantees that Algorithm 4 will return a feasible solution of $\mathcal{SNCE}_{TWO}$ with integer-valued resource acquisition. We also note that the BDH algorithm does not guarantee the convergence to the optimal solution. Therefore, we need to use parameter k to control the termination of the BDH algorithm.

Algorithm 4 - The BDH Algorithm for $\mathcal{SNCE}_{TWO}$

Set $UB^{\mathcal{LB}}_{\text{tmp}} = UB^{\mathcal{UB}}_{\text{tmp}} = 0$, $LB = -\infty$ and $UB = +\infty$.
Create $\mathcal{MASTER}_{\mathcal{LB}}$ and $\mathcal{MASTER}_{\mathcal{UB}}$ with no cuts.
repeat
    Solve $\mathcal{MASTER}_{\mathcal{LB}}$ and obtain the incumbent solution $\mathcal{X}_{\mathcal{LB}}$
    Solve $\mathcal{MASTER}_{\mathcal{UB}}$ and obtain the incumbent solution $\mathcal{X}_{\mathcal{UB}}$
    $LB = Z^*_{\mathcal{LB}}$
    Using $\mathcal{X}_{\mathcal{LB}}$, let $UB^{\mathcal{LB}}_{\text{tmp}} = \sum_{n\in\mathcal{T}} p_n \sum_{(i,j)\in\mathcal{A}} c^u_{ij} x_{nij}$
    Using $\mathcal{X}_{\mathcal{UB}}$, let $UB^{\mathcal{UB}}_{\text{tmp}} = \sum_{n\in\mathcal{T}} p_n \sum_{(i,j)\in\mathcal{A}} c^u_{ij} x_{nij}$
    for $(i,j) \in \mathcal{A}$
        Solve $\mathcal{DSP}_{ij}$ with $\mathcal{X}_{\mathcal{LB}}$ to get Z^L_{ij}. Add (17) to $\mathcal{MASTER}_{\mathcal{LB}}$
        Solve $\mathcal{DSP}_{ij}$ with $\mathcal{X}_{\mathcal{UB}}$ to get Z^U_{ij}. Add (19) to $\mathcal{MASTER}_{\mathcal{UB}}$
    end for
    for $i \in \mathcal{N}$
        Solve $\mathcal{DSP}_i$ with $\mathcal{X}_{\mathcal{LB}}$ to get Z^L_i. Add (18) to $\mathcal{MASTER}_{\mathcal{LB}}$
        Solve $\mathcal{DSP}_i$ with $\mathcal{X}_{\mathcal{UB}}$ to get Z^U_i. Add (20) to $\mathcal{MASTER}_{\mathcal{UB}}$
    end for
    $UB = \text{Min}\left(UB, \sum_{ij} Z^L_{ij} + \sum_i Z^L_i + UB^{\mathcal{LB}}_{\text{tmp}}, \sum_{ij} Z^U_{ij} + \sum_i Z^U_i + UB^{\mathcal{UB}}_{\text{tmp}}\right)$
    GAP $= (UB - LB)/LB$.
until GAP $< \epsilon$ **or** no gap improvement for k iterations.

6 Extension

We can extend all the results in Section 4 and Section 5 to the situation where three sources of capacity, i.e., permanent, spot market, and contract capacities, interact with each other. We assume that the contract capacity will be available in the next period, if a contract has been signed in the current period. Here, we consider the case that the contract length is one period, but our model can capture the case where the contract can be signed for an arbitrary number of periods. The stochastic network capacity expansion model with three sources of capacity can be formulated as follows:

Model $\mathcal{SNCE}_{THREE}$:

$$\text{Min} \sum_{n\in\mathcal{T}} p_n \left[\sum_{(i,j)\in\mathcal{A}} (c^u_{ij}x_{nij} + c^{up}_{nij}u^p_{nij} + c^{us}_{nij}u^s_{nij} + c^{uc}_{nij}u^c_{nij}) \right.$$
$$\left. + \sum_{i\in\mathcal{N}} \left(\sum_{(i,j)\in\delta^+(i)} x_{nij} + c^{vp}_{ni}v^p_{ni} + c^{vs}_{ni}v^s_{ni} + c^{vc}_{ni}v^c_{ni} \right) \right]$$

s.t.

$$\sum_{(j,i)\in\delta^-_i} x_{nji} - \sum_{(i,j)\in\delta^+_i} x_{nij} = d_{ni} \qquad \forall i \in \mathcal{N}, \forall n \in \mathcal{T},$$

$$x_{nij} \le C^u_{ij} \left(\sum_{m\in\bar{\mathcal{P}}(n)} u^p_{mij} + u^s_{nij} + u^c_{a(n)ij} \right) \qquad \forall (i,j) \in \mathcal{A}, \forall n \in \mathcal{T},$$

$$\sum_{(i,j)\in\delta^+_i} x_{nij} \le C^v_i \left(\sum_{m\in\bar{\mathcal{P}}(n)} v^p_{mi} + v^s_{ni} + v^c_{a(n)i} \right) \qquad \forall i \in \mathcal{N}, \forall n \in \mathcal{T},$$

$$x_{nij} \in \mathbb{R}^+ \qquad \forall (i,j) \in \mathcal{A}, \forall n \in \mathcal{T},$$

$$u^p_{nij},\ u^s_{nij},\ u^c_{nij} \in \mathbb{Z}^+ \qquad \forall (i,j) \in \mathcal{A}, \forall n \in \mathcal{T},$$

$$v^p_{ni},\ v^s_{ni},\ v^c_{ni} \in \mathbb{Z}^+ \qquad \forall i \in \mathcal{N}, \forall n \in \mathcal{T}.$$

Similar to $\mathcal{SNCE}_{TWO}$, it can be proved that $\mathcal{SNCE}_{THREE}$ is an NP-hard problem. The LPA algorithm in Section 4 can be used to solve this problem, since $\mathcal{SNCE}_{THREE}$ has the same decomposition structure as $\mathcal{SNCE}_{TWO}$, i.e., it can be decomposed into single resource capacity expansion problems corresponding to nodes and arcs of the network. Each subproblem of $\mathcal{SNCE}_{THREE}$, which includes all three kinds of capacity, can be formulated as:

$$\begin{aligned} \mathcal{SCE}_{THREE}: \quad \text{Min} \quad & \sum_{n\in\mathcal{T}} p_n \left(c^p_n x_n + c^s_n z_n + c^c_n y_n\right) & \\ \text{s.t.} \quad & \sum_{m\in\bar{\mathcal{P}}(n)} x_m + z_n + y_{a(n)} \ge d_n & \forall n \in \mathcal{T}, \\ & x_n, z_n, y_n \in \mathbb{Z}^+ & \forall n \in \mathcal{T}. \end{aligned} \tag{21}$$

Therefore, the LPA algorithm can also be applied to $\mathcal{SNCE}_{THREE}$, provided that in step 2, $\mathcal{SCE}_{THREE}$ can be solved efficiently for all nodes and arcs.

In order to solve $\mathcal{SCE}_{THREE}$, polynomial-time algorithms have been designed and presented in Taghavi and Huang (2014). It is noteworthy that these algorithms and their validity proofs are different from those in the Appendix due to the interaction of three sources of capacity instead of two sources.

For $\mathcal{SNCE}_{THREE}$, we can define $\mathcal{Y} = (\mathbf{y}_n)_{n\in\mathcal{T}} = (\mathbf{u}_n^{\mathbf{p}}, \mathbf{v}_n^{\mathbf{p}}, \mathbf{u}_n^{\mathbf{s}}, \mathbf{v}_n^{\mathbf{s}}, \mathbf{u}_n^{\mathbf{c}}, \mathbf{v}_n^{\mathbf{c}})_{n\in\mathcal{T}}$ where $\mathbf{u}_n^{\mathbf{p}} = (u_{nij}^p)_{(i,j)\in\mathcal{A}}$, $\mathbf{v}_n^{\mathbf{p}} = (v_{ni}^p)_{i\in\mathcal{N}}$, $\mathbf{u}_n^{\mathbf{s}} = (u_{nij}^s)_{(i,j)\in\mathcal{A}}$, $\mathbf{v}_n^{\mathbf{s}} = (v_{ni}^s)_{i\in\mathcal{N}}$, $\mathbf{u}_n^{\mathbf{c}} = (u_{nij}^c)_{(i,j)\in\mathcal{A}}$, and $\mathbf{v}_n^{\mathbf{c}} = (v_{ni}^c)_{i\in\mathcal{N}}$. Then the following results hold:

Theorem 8. *For $\mathcal{SNCE}_{THREE}$, given the same assumptions as in Theorem 6, we have*

$$\lim_{T\to\infty} \frac{f(\mathcal{X}^A, \mathcal{Y}^A) - f(\mathcal{X}^*, \mathcal{Y}^*)}{f(\mathcal{X}^*, \mathcal{Y}^*)} = 0$$

The proof of Theorem 8 is analogous to that of Theorem 6. So, we do not present it here for brevity. Theorem 8 shows that the LPA algorithm keeps its asymptotic convergence property when dealing with three sources of capacity.

By minor changes, the BDH algorithm presented in Section 5 can also be applied to solve the $\mathcal{SNCE}_{THREE}$ problem. We only need to incorporate contract capacity into the capacity constraints of arc and node subproblems $\mathcal{SP}_{(i,j)}$ and $\mathcal{SP}_i$. The rest of the BDH algorithm remain the same for $\mathcal{SNCE}_{THREE}$.

7 Experimental Results

In this section, we present the experimental results for the proposed algorithms in three parts. In Section 7.1, we present the results for solving a single resource capacity expansion problem (8) and investigate the effect of having two sources of capacity versus having three sources of capacity. Section 7.2, considers the efficiency and the asymptotic convergence property of the LPA algorithm. In Section 7.3, we present the experimental results for the BDH algorithm and we compare them with those of the LPA algorithm.

All algorithms have been implemented in C++ using IBM/ILOG CPLEX 12.6 Concert Technology. All experiments were performed on a 2.5GHz Intel Core i5-2520M processor with 8.00 GB of RAM running Microsoft Windows 8.1.

7.1 Single Resource Capacity Expansion

In this section, we present the experiments that have been designed to observe the effect of having multiple source of capacities for the single resource capacity expansion problems $\mathcal{SCE}_{TWO}$ and $\mathcal{SCE}_{THREE}$. Our goal is to show the benefits of having three sources of capacity compared to the case of two sources of capacity. Also, we are interested in the percentages of different capacities in the optimal solution and their impacting factors.

All test instances have been generated as follows. We fix the number of periods (T_f) that a permanent capacity can be used and we randomly generate an overall permanent capacity cost (c^p) for T_f. This implies that the cost of permanent capacity for each period is $\frac{c^p}{T_f}$. Since the permanent capacity will be available for all future periods, for each node $n \in \mathcal{T}$, the permanent capacity cost when we consider T periods is set to be $c^p_n = (T - t_n)\frac{c^p}{T_f}$, where t_n is the period of node n. For spot market capacity, the base cost for node $n \in \mathcal{T}$ is set to be $\frac{c^p}{T_f}$. The base contract capacity cost will be $L\frac{c^p}{T_f}$, where L is the length of the contract.

We also add demand and price variability as the number of periods increases. For permanent capacity, we assumed that the cost will increase by $\%\theta^p$ per period. Spot market cost can also increase by $\%\theta^s$ per period. For contract capacity, we assume that signing a contract will be rewarded a discount of $\%\alpha^c$. Moreover, we assume that the demand follows a Log-normal distribution $\lambda(\mu, \sigma)$. This assumption, that follows other similar works in the literature (Ahmed and Sahinidis (2003); Huang and Ahmed (2009)), guarantees that the generated demands are nonnegative. We randomly generate the demand for the root node (d_1), and assume that μ and σ are increasing as the periods increase. So, for $n \in \mathcal{T}$, we have $d_n = d_1\lambda(\mu + \beta_m t_n, \sigma + \beta_s t_n)$. This indicates that as the number of periods increases, demand variability increases.

To create test instances, we alternated the value of θ^p and α^c among 0, 0.1, and 0.2, and the value of θ^s among 0.1, 0.2, 0.3, 0.4, 0.5. Also, we set $\mu = 0.5$ and $\sigma = 0.25$ and change the value of β_m between 0, 0.1, 0.2 and the value of β_s between 0.2 and 0.5. By changing T from 3 to 7 and B from 2 to 15, we created 15 different scenario trees with sizes from 7 to 5,461. The combination of scenario trees with different parameters led to solving 7,290 instances of single resource capacity expansion problem with both two and three sources of capacity.

Table 3: Percentage of using different types of capacity and improvement in the objective function using $\mathcal{SCE}_{THREE}$ over $\mathcal{SCE}_{TWO}$.

α^c	0.1						0.2					
	$\mathcal{SCE}_{TWO}$		$\mathcal{SCE}_{THREE}$			*Imp.*	$\mathcal{SCE}_{TWO}$		$\mathcal{SCE}_{THREE}$			*Imp.*
θ^s	P	S	P	C	S		P	S	P	C	S	
0.1	21	79	2	39	59	108.47	23	77	3	42	55	126.49
0.2	26	74	3	38	59	100.17	25	75	3	41	56	119.23
0.3	29	71	11	33	56	81.12	28	72	7	40	53	103.43
0.4	32	68	12	35	53	75.49	32	68	11	35	54	93.88
0.5	32	68	15	30	55	76.94	35	65	12	37	51	84.64

We present the following results from our experiments. For each set of problems, we state the percentage of permanent (P) and spot market capacity (S) used in the solution of $\mathcal{SCE}_{TWO}$ model. We also present the percentage of permanent (P), spot market (S), and contract capacity (C) used in the solution of $\mathcal{SCE}_{THREE}$ model. Finally, we present the percentage of improvement ($Imp.$) in the value of the objective function if we have access to three sources of capacity compared to the case with two sources of capacity.

For brevity of exposition, we only report one table (Table 3) for the case in which $\theta^p = 0$ (we observed similar results when $\theta^p = 0.1$ and $\theta^p = 0.2$). In order to test both low and high demand variability ($\beta_m = 0, 0.1, 0.2$ and $\beta_s = 0.2, 0.5$), we solve each one of our 15 instances for all 6 demand combinations. As a result, the percentage of different types of capacity and improvement is the average result for $15 \times 6 = 90$ instances.

From Table 3, it can be observed that as the spot market cost increases, the use of permanent capacity increases in both $\mathcal{SCE}_{TWO}$ and $\mathcal{SCE}_{THREE}$. However, for higher contract discounts, the increase of permanent capacity usage is less. We can also observe that when contract capacity is an option, the objective function can be significantly improved. The reason is that contract capacity is a more flexible option compared to permanent capacity (it only considers limited periods, compared to permanent capacity that considers all remaining periods), and is cheaper compared to spot market capacity.

7.2 Convergence of the LPA Algorithm

For the LPA algorithm, we present the results to: (i) verify the convergence result in Theorem 6, and (ii) show the performance of the LPA algorithm compared to CPLEX MIP solver.

In order to observe the convergence of the LPA algorithm, it is necessary to solve the same problem with scenario trees that are increasing in the number of periods (according to Theorem 6). For this purpose, the instances that we create in this part have a fixed network structure, and a varying scenario tree structure. This enables us to observe the gap returned by the LPA algorithm for different scenario tree sizes that are used to model the same network.

For each problem, we report the number of nodes and arcs in the network, and the number of periods and branches in the scenario tree. We assume that there are three types of nodes in the network: origin, destination and transient, with negative, positive and zero demands respectively. For the same number of nodes in the network, having more transient nodes will result in simpler instances. Thus we report the number of transient nodes in the network as well. We also report the number of variables and constraints for each problem. In order to make the result independent from the parameters, we randomly create 10 instances for each problem. Since for each problem, we keep the same network structure and change the scenario tree periods (while fixing B to 2), $|\mathcal{N}|$, $|\mathcal{A}|$, and the number of transient nodes are fixed for each set of problems. The result in each row of Table 4 is the average results from solving 10 randomly generated instances. In other words, Table 4 summarizes the data resulted from solving 250 randomly generated instances.

From Table 4, we can observe that the LPA algorithm can outperform CPLEX MIP solver in all instances. Also, it can be observed that for each set of problems defined on the same network structure, the optimality gap is decreasing while the number of periods increases. This confirms the result of Theorem 6.

Our experiments for this part is not limited to what we presented in Table 4. We also did experiments for the case where $B = 3$. For all instances of this case, LPA algorithm outperformed CPLEX MIP solver and the convergence to the optimal solution could be observed. Since the results are similar, we do not present them here. Also, note that we generated small enough instances in Table 4 to be solved by CPLEX MIP solver within one hour. Therefore, we designed large-scale instances of the problems that can-

Table 4: LPA algorithm performance for instances of $\mathcal{SNCE}_{TWO}$

$\lvert\mathcal{N}\rvert$	Trans. Nodes	$\lvert\mathcal{A}\rvert$	T	No. of Variables		No. of Constraints	Time(s)		Optimality Gap (%)
				Real	Integer		CPLEX	LPA	
8	4	20	2	60	168	108	0.060	0.019	5.986
			3	140	392	252	0.143	0.025	3.325
			4	300	840	540	1.481	0.048	1.253
			5	620	1,736	1,116	4.599	0.145	0.530
			6	1,260	3,528	2,268	9.732	0.286	0.170
			7	2,540	7,112	4,572	14.556	0.518	0.018
10	6	30	2	90	240	150	0.244	0.055	3.871
			3	210	560	350	0.785	0.089	1.156
			4	450	1,200	750	12.904	0.108	0.663
			5	930	2,480	1,550	67.780	0.222	0.292
			6	1,890	5,040	3,150	91.810	0.430	0.137
			7	3,810	10,160	6,350	106.545	0.710	0.019
15	9	50	2	150	390	240	0.487	0.046	4.230
			3	350	910	560	3.967	0.087	1.283
			4	750	1,950	1,200	24.652	0.146	0.824
			5	1,550	4,030	2,480	71.292	0.672	0.449
			6	3,150	8,190	5,040	484.910	0.819	0.181
			7	6,350	16,510	10,160	501.701	1.181	0.013
20	13	90	2	270	660	390	0.653	0.073	5.281
			3	630	1,540	910	6.596	0.152	3.371
			4	1,350	3,300	1,950	170.175	0.486	1.034
			5	2,790	6,820	4,030	631.230	0.785	0.689
			6	5,670	13,860	8,190	988.627	1.156	0.172
			7	11,430	27,940	16,510	1,469.843	2.394	0.051

Table 5: LPA Vs. BDH algorithm performance for instances of $\mathcal{SNCE}_{TWO}$

$\|\mathcal{N}\|$	$\|\mathcal{A}\|$	T	B	No. of Variables		No. of	Time(s)			Optimality Gap(%)	
				Real	Integer	Constraints	CPLEX	BDH	LPA	BDH	LPA
10	50	3	2	350	840	490	0.08	1.40	0.06	4.777	7.870
30	400	3	2	2,800	6,020	3,220	927.24	262.71	0.82	7.168	4.036
10	50	3	3	650	1,560	910	38.71	6.40	0.07	5.334	4.677
30	400	3	3	5,200	11,180	5,980	3,600.00	429.47	1.42	6.343	4.405
10	50	4	2	750	1,800	1,050	14.60	2.27	0.08	4.544	4.216
30	400	4	2	6,000	12,900	6,900	3,600.00	806.35	1.82	5.742	3.398
10	50	4	3	2,000	4,800	2,800	261.06	10.70	0.16	2.084	3.333
30	400	4	3	16,000	34,400	18,400	3,600.00	1,120.22	3.94	6.065	2.981
10	50	5	2	1,550	3,720	2,170	3,600.00	15.45	0.13	0.859	0.847
30	400	5	2	12,400	26,660	14,260	3,600.00	978.56	2.53	3.768	0.985
10	50	5	3	6,050	14,520	8,470	3,600.00	31.56	0.83	1.191	0.810
30	400	5	3	48,400	104,060	55,660	3,600.00	3,187.03	20.89	1.355	1.647

not be solved by MIP Solver within one hour. In order to find the optimality gap for those problems, we used the optimal objective value of linear relaxation as the lower bound and the feasible solution returned by the LPA algorithm as the upper bound. We observe the same convergence behavior for those large-scale instances and the results are presented in the Appendix.

7.3 LPA versus BDH

In this section, we present the experimental results related to the BDH algorithm. We compare the solution time of CPLEX MIP solver with that of the BDH algorithm. Also, we compare the qualities of gaps provided by LPA and BDH algorithms.

Similar to previous section, we report the number of nodes and arcs in the network, the structure of the scenario tree, and the number of variables and constraints. We solve each instance using CPLEX MIP solver, BDH algorithm and LPA algorithm and report the solution time and the gaps. For these instances, the number of transient nodes is 50% of the total nodes in the network. The results are presented in Table 5. The solution time and gap presented for each row are the averages of 5 randomly generated instances.

There are several observations about the results presented in Table 5.

For some instances, CPLEX MIP solver cannot solve the problem within an hour and we report this in the table with "3,600.00" as the solution time. While both LPA and BDH algorithms outperform CPLEX MIP solver in solution time, LPA algorithm has better solution times compared to BDH algorithm. Interestingly, the BDH algorithm can provide tighter bound in some cases compared to LPA algorithm (in almost 19% of total 150 solved instances). It is noteworthy that unlike the LPA algorithm which is asymptotically convergent, there is no analytical conclusion for the BDH algorithm. Therefore, if the number of scenario tree periods is large, the LPA algorithm is likely to outperform the BDH algorithm in almost all instances. For other instances, it would worth spending more time to find tighter gaps using the BDH algorithm. On the other hand, in the experiments we find that the BDH algorithm is more efficient compared to the LPA algorithm in memory usage. With our current computational setting, there are instances that can be solved by the BDH algorithm, and cannot be solved by the LPA algorithm due to out of memory exception.

In the $\mathcal{SNCE}_{TWO}$ and $\mathcal{SNCE}_{THREE}$ problems, a feasible solution can be obtained by solving an LP relaxation of the problem and rounding all the integer variables up. One might be interested to know the quality of the gap resulting from these upper and lower bounds compared to the gap provided by the BDH algorithm. We conducted some experiments and found out that although the BDH algorithm does not guarantee providing tighter gaps compared to LP, in almost 70% of instances (104 out of 150), it provided gaps tighter than the LP gap. The average gap improvement for these 150 instances is 43.05%. Moreover, there are instances for which, the BDH algorithm can provide the optimal solution of the original problem, indicating that one of the upper or lower bounds returned by it is exactly equal to the objective value of the optimal solution.

8 Conclusions

In this paper, we proposed a general model of stochastic network capacity expansion in the presence of permanent, spot market, and contract capacities. The problem has been modeled in the context of a min-cost network flow problem using multi-stage stochastic programming approach. This general framework can be applied to a wide range of network applications.

These models are large scale mixed-integer programming models which

are notoriously difficult to solve. Therefore, we explored the decomposable structure, and defined subproblems corresponding to each resource (e.g., node, arc), and proposed polynomial-time algorithms to solve them. Then, we incorporated the subproblem solution techniques into an LP based approximation algorithm (LPA), which is not only computationally efficient, but also can be shown to be asymptotically convergent. We also developed a heuristic algorithm (BDH) based on Benders decomposition method that can provide tight bounds for our models. In some cases, the gaps provided by the BDH algorithm are shown to be tighter compared to LPA gaps. Finally, BDH algorithm seems to be more efficient than LPA when it comes to memory usage. Computational results show that the proposed approaches are much more efficient than the commercial solver CPLEX.

The problem that we studied is defined on a min-cost network flow problem. One direction for future study is to change the underlying network flow problem to a multi-commodity flow problem. In this case, the LPA algorithm will fail because multiple mixed-integer programming subproblems must be solved. Therefore, a different solution methodology must be developed. Another direction is to develop exact algorithms for the proposed model, especially when the planning horizon is short.

References

Ahmed, S. and Sahinidis, N. V. (2003). An approximation scheme for stochastic integer programs arising in capacity expansion. *Operations Research*, 51(3):461–471.

Ahuja, R. K., Batra, J. L., Gupta, S. K., and Punnen, A. P. (1996). Optimal expansion of capacitated transshipment networks. *European Journal of Operational Research*, 89(1):176–184.

Atamtürk, A. and Hochbaum, D. S. (2001). Capacity acquisition, subcontracting, and lot sizing. *Management Science*, 47(8):1081–1100.

Atamtürk, A. and Zhang, M. (2007). Two-stage robust network flow and design under demand uncertainty. *Operations Research*, 55(4):662–673.

Balakrishnan, A., Magnanti, T. L., Shulman, A., and Wong, R. T. (1991). Models for planning capacity expansion in local access telecommunication networks. *Annals of Operations Research*, 33(4):237–284.

Bansal, P. P. and Jacobsen, S. E. (1975). An algorithm for optimizing network flow capacity under economies of scale. *Journal of Optimization Theory and Applications*, 15(5):565–586.

Bean, J. C., Higle, J. L., and Smith, R. L. (1992). Capacity expansion under stochastic demands. *Operations Research*, 40(3-supplement-2):S210–S216.

Berman, O. and Ganz, Z. (1994). The capacity expansion problem in the service industry. *Computers & Operations Research*, 21(5):557–572.

Bhatnagar, S., Fernández-Gaucherand, E., Fu, M. C., He, Y., and Marcus, S. I. (1998). A Markov decision process model for capacity expansion and allocation (I). *IEEE Conference On Decision and Control*, 2(1):1380–1385.

Bienstock, D., Chopra, S., Günlük, O., and Tsai, C. Y. (1998). Minimum cost capacity installation for multicommodity network flows. *Mathematical Programming*, 81(2):177–199.

Bienstock, D. and Günlük, O. (1996). Capacitated network design-polyhedral structure and computation. *INFORMS Journal on Computing*, 8(3):243–259.

Christofides, N. and Brooker, P. (1974). Optimal expansion of an existing network. *Mathematical Programming*, 6(1):197–211.

Doulliez, P. J. and Rao, M. R. (1971). Capacity of a network with increasing demands and arcs subject to failure. *Operations Research*, 19(4):905–915.

Doulliez, P. J. and Rao, M. R. (1975). Optimal network capacity planning: A shortest-path scheme. *Operations Research*, 23(4):810–818.

Erkoc, M. and Wu, S. D. (2009). Managing high-tech capacity expansion via reservation contracts. *Production and Operations Management*, 14(2):232–251.

Fulkerson, D. R. (1959). Increasing the capacity of a network: The parametric budget problem. *Management Science*, 5(4):472–483.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company.

Günlük, O. (1999). A branch-and-cut algorithm for capacitated network design problems. *Mathematical Programming*, 86(1):17–39.

Han, X. and Makino, K. (2010). *Approximation and Online Algorithms*. Springer Berlin Heidelberg, Berlin, Heidelberg.

Hsu, N. S., Cheng, W. C., Cheng, W. M., Wei, C. C., and Yeh, W. G. (2008). Optimization and capacity expansion of a water distribution system. *Advances in Water Resources*, 31(5):776–786.

Huang, K. and Ahmed, S. (2009). The value of multistage stochastic programming in capacity planning under uncertainty. *Operations Research*, 57(4):893–904.

Inderfurth, K. and Kelle, P. (2011). Capacity reservation under spot market price uncertainty. *International Journal of Production Economics*, 133(1):272–279.

Inderfurth, K., Kelle, P., and Kleber, R. (2013). Dual sourcing using capacity reservation and spot market: Optimal procurement policy and heuristic parameter determination. *European Journal of Operational Research*, 225(2):298–309.

Karoonsoontawong, A. (2010). Integrated network capacity expansion and traffic signal optimization problem: robust bi-level dynamic formulation. *Networks and Spatial Economics*, 10(4):525–550.

Lee, C. and Kang, H. (2000). Cell planning with capacity expansion in mobile communications: a tabu search approach. *IEEE Transactions on Vehicular Technology*, 49(5):1678–1691.

Lee, D. H., Lim, S. K., Lee, G. C., Jun, H. B., and Kim, Y. D. (1997). Multi-period part selection and loading problems in flexible manufacturing systems. *Computers & Industrial Engineering*, 33(3):541–544.

Leung, J. M. Y., Magnanti, T. L., and Singhal, V. (1990). Routing in point-to-point delivery systems: formulations and solution heuristics. *Transportation Science*, 24(4):245–260.

Levin, Y., Nediak, M., and Topaloglu, H. (2012). Cargo capacity management with allotments and spot market demand. *Operations Research*, 60(2):351–365.

Liu, J., Ahuja, R. K., and Sahin, G. (2007). Optimal network configuration and capacity expansion of railroads. *Journal of the Operational Research Society*, 59(7):911–920.

Logendran, R. and Puvanunt, V. (1997). Duplication of machines and subcontracting of parts in the presence of alternative cell locations. *Computers & Industrial Engineering*, 33(1):235–238.

Logendran, R. and Ramakrishna, P. (1997). A methodology for simultaneously dealing with machine duplication and part subcontracting in cellular manufacturing systems. *Computers & Operations Research*, 24(2):97–116.

Luss, H. (1984). Capacity expansion planning for a single facility product line. *European Journal of Operational Research*, 18(1):27–34.

Magnanti, T. L., Mirchandani, P., and Vachani, R. (1995). Modeling and solving the two-facility capacitated network loading problem. *Operations Research*, 43(1):142–157.

Magnanti, T. L. and Wong, R. T. (1981). Accelerating Benders Decomposition: Algorithmic Enhancement and Model Selection Criteria. *Operations Research*, 29(3):464–484.

Magnanti, T. L. and Wong, R. T. (1984). Network design and transportation planning: Models and algorithms. *Transportation Science*, 18(1):1–55.

Marín, A. and Jaramillo, P. (2008). Urban rapid transit network capacity expansion. *European Journal of Operational Research*, 191(1):45–60.

McDaniel, D. and Devine, M. (1977). A modified Benders' partitioning algorithm for mixed integer programming. *Management Science*, 24(3):312–319.

Mudchanatongsuk, S., Ordóñez, F., and Liu, J. (2007). Robust solutions for network design under transportation cost and demand uncertainty. *Journal of the Operational Research Society*, 59(5):652–662.

Ordóñez, F. and Zhao, J. (2007). Robust capacity expansion of network flows. *Networks*, 50(2):136–145.

Oren, S., Jiang, J., Wu, D., Kleindorfer, P. R., and Sun, Y. (2005). Optimal capacity expansion in the presence of capacity options. *Decision Support Systems*, 40(3):553–561.

Pimentel, B. S., Mateus, G. R., and Almeida, F. A. (2013). Stochastic capacity planning and dynamic network design. *International Journal of Production Economics*, 145(1):139–149.

Pratikakis, N. E. (2010). Strategic capacity decision-making in a stochastic manufacturing environment using real-time approximate dynamic programming. *Naval Research Logistics*, 57(3):211–224.

Saharidis, G. K. D. and Ierapetritou, M. G. (2010). Improving benders decomposition using maximum feasible subsystem (MFS) cut generation strategy. *Computers & Chemical Engineering*, 34(8):1237–1245.

Sen, S., Doverspike, R. D., and Cosares, S. (1994). Network planning with random demand. *Telecommunication Systems*, 3(1):11–30.

Taghavi, M. and Huang, K. (2014). Stochastic capacity expansion with multiple sources of capacity. *Operations Research Letters*, 42(4):263–267.

Van Mieghem, J. A. (2003). Commissioned paper: Capacity management, investment, and hedging: Review and recent developments. *Manufacturing & Service Operations Management*, 5(4):269–302.

Wolsey, L. A. and Nemhauser, G. L. (1988). *Integer and combinatorial optimization*. Wiley-Interscience, New York, 18 edition.

Zhang, F., Roundy, R. O., Cakanyildirim, M., and Huh, W. T. (2004). Optimal capacity expansion for multi-product, multi-machine manufacturing systems with stochastic demand. *IIE Transactions*, 36(1):23–36.

Appendix

Proof of Theorem 1

Theorem 1. *The stochastic program $\mathcal{SNCE}_{TWO}$ and its deterministic counterpart are NP-hard.*

Proof. We show that any instance of the NP-hard integer knapsack problem (Garey and Johnson (1979)) with $|\mathcal{N}| + |\mathcal{A}|$ items can be polynomially transformed to a single period instance of the deterministic network capacity expansion problem, which is just a single node scenario instance of the stochastic model $\mathcal{SNCE}_{TWO}$. Suppose the scenario tree has one node, so we can drop the index n in the model. Also, let u^p_{nij} and v^p_{ni} be zero. Now, from (2), we have $\sum_{(j,i)\in\delta_i^-} x_{ji} - d_i = \sum_{(i,j)\in\delta_i^+} x_{ij}$. If we insert this in (4), we will have $\sum_{(j,i)\in\delta_i^-} x_{ji} - d_i \leq C^v_i v^s_i$, or $C^v_i v^s_i - \sum_{(j,i)\in\delta_i^-} x_{ji} \geq -d_i$. Also, according to (3), we have $x_{ij} \leq C^u_{ij} u^s_{ij}$ or $\sum_{(i,j)\in\delta_i^-} x_{ij} \leq \sum_{(i,j)\in\delta_i^-} C^u_{ij} u^s_{ij}$, or $-\sum_{(j,i)\in\delta_i^-} x_{ji} \leq \sum_{(i,j)\in\delta_i^-} C^u_{ij} u^s_{ij}$. Therefore, $C^v_i v^s_i + \sum_{(i,j)\in\delta_i^-} C^u_{ij} u^s_{ij} \geq -d_i$. Now assume that in the network, there is only one origin (node 1) and all other nodes are destinations. So, for all nodes i other than node 1, we have $v^s_i = \lceil \frac{d_i}{C^v_i} \rceil$, where $\lceil x \rceil$ is the closest integer value greater than or equal to x; and the rest of the problem can be stated as follows:

$$\begin{aligned} \text{Min} \quad & \sum_{(1,j)\in\mathcal{A}} c^{us}_{1j} u^s_{1j} + c^{vs}_1 v^s_1 \\ \text{s.t.} \quad & C^v_1 v^s_1 + \sum_{(1,j)\in\delta_1^-} C^u_{1j} u^s_{1j} \geq -d_1 \\ & u^s_{1j}, v^s_1 \in \mathbb{Z}^+, \qquad \forall j \in \mathcal{N} \end{aligned}$$

which can be seen as the integer knapsack problem presented by Garey and Johnson (1979). Note that minimization knapsack problem can be transfered into an equivalent maximization knapsack problem in polynomial time (Han and Makino (2010)). □

Proof of Theorem 2

Theorem 2. *If the demands are integer-valued, the LP relaxation of model $\mathcal{SCE}-2$ will yield integral optimal solutions.*

Proof. Suppose A is the left hand side matrix of the model $\mathcal{SCE}-2$. Note that A is a 0-1 matrix and each row of it corresponds to a node in the scenario tree. Consider the z_n columns first. There is only a single 1 in each column and all other entries are zero. Now, consider columns of x_n. Entry of row i and column j is 1 only if $i \in \bar{\mathcal{T}}(j)$. Now, we reorder the constraints based on depth-first search method (for each column j, the rows corresponding

to $\bar{\mathcal{T}}(j)$ are consecutive, and row $i_1 \in \bar{\mathcal{T}}(j)$ appears before row $i_2 \in \bar{\mathcal{T}}(j)$ if $i_2 \in \mathcal{T}(i_1)$). This new ordering guarantees that in each column, the 1s appear consecutively. Therefore, A is an interval matrix which is totally unimodular.(Wolsey and Nemhauser (1988)) □

Proof of Lemmas

In order to prove Theorem 3, we need to present and prove some lemmas first. Lemma 1 presents the dual problem optimality criteria:

Lemma 1. *A solution $\pi = (\pi_1, \pi_2, \cdots, \pi_n)$ is dual feasible if and only if:*

$$0 \leq \pi_n \leq \text{Min} \left\{ c_n^s, \underset{m \in \bar{\mathcal{P}}(n)}{\text{Min}} \left\{ c_m^p - \sum_{k \in \bar{\mathcal{T}}(m) \setminus \{n\}} \pi_k \right\} \right\}. \tag{22}$$

When π is optimal, the second inequality is tight.

Proof. Non-negativity of π_n and $\pi_n \leq c_n^s$ are guaranteed by constraints from dual problem (9). Also, note that for each n, all constraints in which π_n appears are corresponding to $\bar{\mathcal{P}}(n)$. Therefore, for all $m \in \bar{\mathcal{P}}(n)$, we have$\sum_{k \in \bar{\mathcal{T}}(m)} \pi_k \leq c_m^p$ or $\pi_n + \sum_{k \in \bar{\mathcal{T}}(m) \setminus \{n\}} \pi_k \leq c_m^p$. So, for all $m \in \bar{\mathcal{P}}(n)$, we require that $\pi_n \leq c_m^p - \sum_{k \in \bar{\mathcal{T}}(m) \setminus \{n\}} \pi_k$ or

$$\pi_n \leq \underset{m \in \bar{\mathcal{P}}(n)}{\text{Min}} \left\{ c_m^p - \sum_{k \in \bar{\mathcal{T}}(m) \setminus \{n\}} \pi_k \right\}$$

Now suppose in the optimal solution, the second inequality is not tight. As a result, we can increase π_n to get a new feasible solution with greater objective value, which is a contradiction. □

Lemma 2 shows a property of the primal solution resulting from Algorithm 1:

Lemma 2. *In the primal algorithm, if $x_n^* > 0$, there exists $k \in \bar{\mathcal{T}}(n)$, such that $\sum_{m \in \mathcal{P}(n)} x_m^* = d_k$.*

Proof. The scenario tree has T periods and we use mathematical induction from period T to 1 to prove the statement. Consider the steps in Algorithm 1.

The initial solution requires that $x_n^* = 0$ and $z_n^* = d_n$ for all $n \in \mathcal{T}$. The statement is true for the initial solution because $x_n^* = 0$ for all n. For a non-leaf nodes k in period $T-1$, if we shift up capacity from nodes in $\mathcal{A}(k)$ to k as permanent capacity, then $x_k^* = \Delta_k > 0$. Since Δ_k is the minimum demand among all nodes in $\mathcal{A}(k)$, the statement holds. If it is still more economical to shift up capacity from nodes in $\mathcal{A}(k)$ to node k, the next value of x_k^* will be equal to the second smallest demand in $\mathcal{A}(k)$. When this procedure stops for node k, the capacity accumulated in node k is equal to the demand of a node in $\mathcal{A}(k)$, so the statement holds. Now consider any non-leaf nodes k in any period before $t < T-1$. Considering the definition of Δ_k and the structure of initial solution, if Δ_k takes the minimum value among spot market capacities, it will be equal to the demand of a direct descendant node. If Δ_k takes the minimum value among permanent capacities, the values are equal to the demand of a future node due to mathematical induction assumption. Therefore, the statement holds for the whole scenario tree. □

In Lemma 3, we show several properties of the dual solutions resulting from Algorithm 2. To facilitate our following proofs, we define m_k for each node k as: $m_k = \text{argmin}_{n \in \bar{\mathcal{P}}(k)} \left\{ c_n^{k-1} \right\}$. When $c_n^{k-1} = c_m^{k-1}$, we let $m_k = m$ if $t_m < t_n$.

Lemma 3. . *In the dual algorithm, the following results hold:*

(a) If $\pi_k^* = c_{m_k}^{k-1}$, *then* $\sum_{m \in \bar{\mathcal{T}}(m_k)} \pi_m^* = c_{m_k}^p$, *and* $\pi_i^* = 0$, $\forall i \in \bar{\mathcal{T}}(m_k)$ *with* $i > k$;

(b) If $\pi_k^* = c_{m_k}^{k-1}$, *then* $\sum_{m \in \bar{\mathcal{T}}(i), m<k} \pi_m^* < c_i^p$, $\forall i \in \bar{\mathcal{P}}(m_k)$.

(c) If $\pi_k^* = c_k^s$, *then* $\sum_{m \in \bar{\mathcal{T}}(i), m<k} \pi_m^* < c_i^p$, $\forall i \in \bar{\mathcal{P}}(k)$.

Proof. (a). Let us define $\mathcal{U}$, $\mathcal{V}$, and $\mathcal{W}$ as follows:

$$\begin{aligned} \mathcal{U} &= \left\{ i \ : \ i \in \bar{\mathcal{T}}(m_k),\ i < k,\ \pi_i^* = \underset{m \in \bar{\mathcal{P}}(i)}{\text{Min}} \{ c_m^{i-1} \} \right\} \\ \mathcal{V} &= \left\{ i \ : \ i \in \bar{\mathcal{T}}(m_k),\ i < k,\ \pi_i^* = c_i^s \right\} \\ \mathcal{W} &= \left\{ i \ : \ i \in \bar{\mathcal{T}}(m_k),\ i > k \right\} \end{aligned}$$

According to the step 4 of Algorithm 2, when we consider node k, we have $\pi_k^* = c_{m_k}^{k-1} = c_{m_k}^p - \sum_{j \in \bar{\mathcal{T}}(m_k), j<k} \pi_j^* = c_{m_k}^p - \sum_{i \in \mathcal{U}} \pi_i^* - \sum_{i \in \mathcal{V}} \pi_i^*$. Consider the dual constraint: $\sum_{m \in \bar{\mathcal{T}}(m_k)} \pi_m^* = \pi_k^* + \sum_{i \in \mathcal{U}} \pi_i^* + \sum_{i \in \mathcal{V}} \pi_i^* + \sum_{i \in \mathcal{W}} \pi_i^* =$

$c^p_{m_k} + \sum_{i\in\mathcal{W}} \pi^*_i \le c^p_{m_k}$. This implies that $\sum_{i\in\mathcal{W}} \pi^*_i = 0$ and since $\pi^*_i \ge 0$ for all i, we must have $\pi^*_i = 0$ for each node $i \in \mathcal{W}$. This also shows that $\sum_{m\in\bar{\mathcal{T}}(m_k)} \pi^*_m = c^p_{m_k}$ which completes the proof for (a).

(b). Suppose $i \in \bar{\mathcal{P}}(m_k)$. According to Algorithm 2, when we consider node k, we have $c^{k-1}_i = c^p_i - \sum_{m\in\bar{\mathcal{T}}(i),m<k} \pi^*_m$ or $\sum_{m\in\bar{\mathcal{T}}(i),m<k} \pi^*_m = c^p_i - c^{k-1}_i$. Since $\pi^*_k = c^{k-1}_{m_k}$, based on the definition of m_k, we have $c^{k-1}_i > c^{k-1}_{m_k} \ge 0$. Therefore, $\sum_{m\in\bar{\mathcal{T}}(i),m<k} \pi^*_m < c^p_i$.

(c). According to Algorithm 2, for any node $i \in \bar{\mathcal{P}}(k)$, we can write $c^{k-1}_i = c^p_i - \sum_{m\in\bar{\mathcal{T}}(i),m<k} \pi^*_m$ or $\sum_{m\in\bar{\mathcal{T}}(i),m<k} \pi^*_m = c^p_i - c^{k-1}_i$. If $c^{k-1}_i = 0$, it requires that $\pi^*_k = 0$ which contradicts $\pi^*_k = c^s_k > 0$. So, $\sum_{m\in\bar{\mathcal{T}}(i),m<k} \pi^*_m < c^p_i$. □

Lemma 4 shows a property of the dual solution resulting from Algorithm 2. To facilitate our proof, we define sets $\mathcal{U}^k_i$ and $\mathcal{V}^k_i$ for node i at iteration k of the dual algorithm:

$$\mathcal{U}^k_i = \left\{ m_j \left| \begin{array}{c} m_j \in \mathcal{T}(i), j<k,\ \pi^*_j = c^{j-1}_{m_j} \\ \text{and there does not exist } l<k \text{ and } l>j \text{ such that} \\ \pi^*_l = c^{l-1}_{m_l},\ m_l \in \mathcal{T}(i),\ \text{and } \mathcal{T}(m_l) \supset \mathcal{T}(m_j) \end{array} \right. \right\}$$

$$\mathcal{V}^k_i = \left\{ j \left| \begin{array}{c} j \in \bar{\mathcal{T}}(i),\ j<k,\ \pi^*_j = c^s_j \\ \text{and there does not exist } l<k \text{ and } l>j \text{ such that} \\ \pi^*_l = c^{l-1}_{m_l},\ m_l \in \mathcal{T}(i),\ \text{and } j \in \mathcal{T}(m_l) \end{array} \right. \right\}$$

These sets correspond to all of the capacities that could not be shifted up to node m_k in the primal algorithm.

Lemma 4. *In the dual algorithm,*

$$c^{k-1}_{m_k} = c^p_{m_k} - \sum_{m_j\in\mathcal{U}^k_{m_k}} c^p_{m_j} - \sum_{j\in\mathcal{V}^k_{m_k}} c^s_j$$

Proof. Consider node i such that there does not exist $j<k$ and $\pi^*_j = cp^{j-1}_{m_j}$ and $i \in \bar{\mathcal{T}}(m_j)$. Note that for any $j<k$, if $j \in \bar{\mathcal{T}}(i)$, then either $j \in \mathcal{V}^k_i$ or $m_j \in \mathcal{U}^k_i$, or there exists $j<l<k$ such that $\pi^*_l = cp^{l-1}_{m_l}$ and $m_l \in \mathcal{T}(i)$ and $j \in \bar{\mathcal{T}}(m_l)$. Therefore, for node i, we have

$$c^{k-1}_i = c^p_i - \sum_{j\in\bar{\mathcal{T}}(i),j<k} \pi^*_j = c^p_i - \sum_{m_j\in\mathcal{U}^k_i} c^p_{m_j} - \sum_{j\in\mathcal{V}^k_i} c^s_j$$

The conclusion holds when $i = m_k$. □

Finally, we connect the primal solution and the dual solution in Lemma 5 and 6, corresponding to the different sources of capacity in the dual solutions:

Lemma 5. *In the dual algorithm, if* $\pi_k^* = c_{m_k}^{k-1}$*, then:*

(a) $z_k^* = 0$*, and* $x_{m_k}^* > 0$ *;*

(b) $\sum_{m\in\mathcal{P}(m_k)} x_m^* \geq d_k$ *;*

(c) For all $i \in \bar{\mathcal{P}}(m_k)$*,* $\sum_{m\in\mathcal{P}(i)} x_m^* < d_k$*.*

Proof. (a). Since $\pi_k^* = c_{m_k}^{k-1}$, it requires that $c_{m_k}^{k-1} < c_k^s$. On the other hand, from Lemma 4, we have $c_{m_k}^{k-1} = c_{m_k}^p - \sum_{m_j\in\mathcal{U}_{m_k}^k} c_{m_j}^p - \sum_{j\in\mathcal{V}_{m_k}^k} c_j^s$ which implies that $c_{m_k}^p < c_k^s + \sum_{m_j\in\mathcal{U}_{m_k}^k} c_{m_j}^p + \sum_{j\in\mathcal{V}_{m_k}^k} c_j^s$. This means that Algorithm 1 will completely shift up the spot market capacity of node k (if any) to m_k, i.e., $z_k^* = 0$.

Next, we need to ensure that the capacity accumulated at node m_k will not be shifted up completely, i.e., $x_{m_k}^* > 0$. Note that for all $i \in \bar{\mathcal{P}}(m_k)$, $c_i^{k-1} = c_i^p - \sum_{m_j\in\mathcal{U}_i^k} c_{m_j}^p - \sum_{j\in\mathcal{V}_i^k} c_j^s$. Since $c_{m_k}^{k-1} < c_i^{k-1}$, we have $c_{m_k}^p - \sum_{m_j\in\mathcal{U}_{m_k}^k} c_{m_j}^p - \sum_{j\in\mathcal{V}_{m_k}^k} c_j^s < c_i^p - \sum_{m_j\in\mathcal{U}_i^k} c_{m_j}^p - \sum_{j\in\mathcal{V}_i^k} c_j^s$, which implies that $c_i^p > c_{m_k}^p + \sum_{m_j\in\mathcal{U}_i^k\setminus\mathcal{U}_{m_k}^k} c_{m_j}^p + \sum_{j\in\mathcal{V}_i^k\setminus\mathcal{V}_{m_k}^k} c_j^s$ for all $i \in \bar{\mathcal{P}}(m_k)$. According to Algorithm 1, this means that the capacity in m_k cannot be shifted up completely anymore. So $x_{m_k}^* > 0$.

(b). To the contrary, Suppose that $\sum_{m\in\mathcal{P}(m_k)} x_m^* < d_k$. Since $z_k^* = 0$, we can conclude that there is a node $j \in \bar{\mathcal{P}}(k) \setminus \mathcal{P}(m_k)$ with $x_j^* > 0$. However, in (a), we proved that Algorithm 1 will shift up any capacity of nodes in $\bar{\mathcal{P}}(k) \setminus \mathcal{P}(m_k)$ to node m_k. So, for all $j \in \bar{\mathcal{P}}(k) \setminus \mathcal{P}(m_k)$, x_j^* must be equal to zero, which is a contradiction.

(c). To the contrary, assume that there exists $i \in \bar{\mathcal{P}}(m_k)$ such that $\sum_{m\in\mathcal{P}(i)} x_m^* \geq d_k$. Without loss of generality, assume that i is such a node with the largest t_i. Then in the primal algorithm, we must have $c_i^p \leq \sum_{m_j\in\mathcal{U}_i^k} c_{m_j}^p + \sum_{j\in\mathcal{V}_i^k} c_j^s$. This implies that node $i = m_k$, which is a contradiction. □

Lemma 6. *In the dual algorithm, if* $\pi_k^* = c_k^s$*, then:*

(a) $z_k^* > 0$ *;*

(b) $\sum_{m\in\bar{\mathcal{P}}(k)} x_m^* + z_k^* = d_k$*, and* $\sum_{m\in\bar{\mathcal{P}}(k)} x_m^* < d_k$ *;*

Proof. (a). Since $\pi_k^* = c_k^s$, we can conclude that for all $i \in \bar{\mathcal{P}}(k)$, $c_k^s < c_i^{k-1}$. According to Lemma 4, $c_i^{k-1} = c_i^p - \sum_{m_j \in \mathcal{U}_i^k} c_{m_j}^p - \sum_{j \in \mathcal{V}_i^k} c_j^s$. So, $c_i^p > c_k^s + \sum_{m_j \in \mathcal{U}_i^k} c_{m_j}^p + \sum_{j \in \mathcal{V}_i^k} c_j^s$ which implies that no capacity will be shifted up from node k to nodes $i \in \bar{\mathcal{P}}(k)$ as permanent capacity according to Algorithm 1. Since the initialization of Algorithm 1 enforces all z_n to be positive, we can conclude that $z_k^* > 0$.

(b). To the contrary, assume that $\sum_{m \in \bar{\mathcal{P}}(k)} x_m^* + z_k^* > d_k$. Since $z_k^* > 0$, we can decrease z_k^* to make the constraint tight. This will give us a new feasible solution with less objective value, which contradicts optimality. Then the second half of the claim follows. □

Proof of Theorem 3

Theorem 3. *The solution* $(x^*, z^*) = (x_1^*, x_2^*, \cdots, x_N^*, z_1^*, z_2^*, \cdots, z_N^*)$ *returned by primal algorithm and the solution* $\pi^* = (\pi_1^*, \pi_2^*, \cdots, \pi_N^*)$ *returned by dual algorithm are optimal.*

Proof. First, we show that (x^*, z^*) and π^* are both feasible solutions of primal and dual problem, respectively. In the primal algorithm, we start with a feasible solution and in each iteration, we shift up Δ_k capacity which preserve feasibility. The feasibility of π^* is guaranteed by Lemma 1.
Next, we need to prove the complementary slackness for primal and dual solutions:

$$\pi_n^* > 0 \implies \sum_{m \in \bar{\mathcal{P}}(n)} x_m^* + z_n^* = d_n \tag{23}$$

$$\sum_{m \in \bar{\mathcal{T}}(n)} \pi_m^* < c_n^p \implies x_n^* = 0 \tag{24}$$

$$\pi_n^* < c_n^s \implies z_n^* = 0 \tag{25}$$

Assume that we are using the dual indexing system. Suppose that $\pi_n^* > 0$. To prove (23), two cases are Considered:

Case 1: If $\pi_n^* = c_n^s$, Lemma 6(b) guarantees that $\sum_{m \in \bar{\mathcal{P}}(n)} x_m^* + z_n^* = d_n$.

Case 2: If $\pi_n^* = c_{m_n}^{n-1}$, to the contrary, assume that $\sum_{m \in \bar{\mathcal{P}}(n)} x_m^* + z_n^* > d_n$. According to Lemma 5(a), $z_k^* = 0$ and $x_{m_n}^* > 0$. Now, according to Lemma 2, there is a node $k \in \bar{\mathcal{T}}(m_n)$ such that $\sum_{m \in \mathcal{P}(m_n)} x_m^* = d_k$, which implies that $d_k > d_n$ or $k < n$. Now, consider the optimal dual variable of node k, i.e. π_k^*. If $\pi_k^* = c_k^s$, Lemma 6(b) requires that $\sum_{m \in \bar{\mathcal{P}}(k)} x_m^* < d_k$, which

contradicts $\sum_{m\in\mathcal{P}(m_n)} x^*_m = d_k$, because $k \in \bar{\mathcal{T}}(m_n)$. If $\pi^*_k = c^{k-1}_{m_k}$, when $m_k \in \bar{\mathcal{T}}(m_n)$, Lemma 5(c) requires that $\sum_{m\in\mathcal{P}(m_n)} x^*_m < d_k$ which contradicts $\sum_{m\in\mathcal{P}(m_n)} x^*_m = d_k$; when $m_k \in \mathcal{P}(m_n)$, then since $n > k$ and $n \in \bar{\mathcal{T}}(m_k)$, Lemma 3(a) requires that $\pi^*_n = 0$ which contradicts $\pi^*_n > 0$. Case $\sum_{m\in\bar{\mathcal{P}}(n)} x^*_m + z^*_n < d_n$ is impossible, because it violates primal feasibility. Therefore,(23) holds.

To prove (24), we again use contradiction. Suppose that $\sum_{m\in\bar{\mathcal{T}}(n)} \pi^*_m < c^p_n$ and $x^*_n > 0$ simultaneously. According to Lemma 2, we can find a $k \in \bar{\mathcal{T}}(n)$, such that $\sum_{m\in\mathcal{P}(n)} x^*_m = d_k$. Now, consider the optimal dual value of k, i.e., π^*_k:

Case 1: $\pi^*_k = c^s_k$. Lemma 6(b) implies that $\sum_{m\in\bar{\mathcal{P}}(k)} x^*_m < d_k$ which contradicts $\sum_{m\in\mathcal{P}(n)} x^*_m = d_k$, since $\mathcal{P}(n) \subseteq \bar{\mathcal{P}}(k)$.

Case 2: $\pi^*_k = c^{k-1}_{m_k}$. If $m_k \in \bar{\mathcal{T}}(n)$, then $n \in \bar{\mathcal{P}}(m_k)$, and Lemma 5(c) requires that $\sum_{m\in\mathcal{P}(n)} x^*_m < d_k$ which contradicts $\sum_{m\in\mathcal{P}(n)} x^*_m = d_k$. If $m_k \in \bar{\mathcal{P}}(n)$, according to Lemma 5(b), we have $\sum_{m\in\mathcal{P}(m_k)} x^*_m \geq d_k$. However, since $\sum_{m\in\mathcal{P}(n)} x^*_m = d_k$ and $n \in \bar{\mathcal{T}}(m_k)$, it requires that $x^*_n = 0$ which is a contradiction. So, the only remaining possibility is $m_k = n$. In this case, Lemma 3(a) requires that $\sum_{m\in\bar{\mathcal{T}}(m_k)} \pi^*_m = c^p_{m_k}$. Since $m_k = n$, then $\sum_{m\in\bar{\mathcal{T}}(n)} \pi^*_m = c^p_n$ which contradicts $\sum_{m\in\bar{\mathcal{T}}(n)} \pi^*_m < c^p_n$. Therefore, (24) holds.

Finally, When $\pi^*_n < c^s_n$, we have a node m_n such that $\pi^*_n = c^{n-1}_{m_n}$. According to Lemma 5(a), we have $z^*_n = 0$. Therefore, (25) holds. □

Proof of Theorem 4

Theorem 4. *The complexity of the primal algorithm is $\mathcal{O}(N^2)$ and the complexity of the dual algorithm is $\mathcal{O}(N \log N \log(\log N))$.*

Proof. In Primal algorithm , the initialization step needs at most N operations. Computing each of $\mathcal{A}(k)$, s_k, and Δ_k requires at most N operations. The external **while** (in step 3) will run at most n times. Each iteration of the internal **while** (in step 5) will deplete the capacity for one node and the variable adjustments will be of the complexity of at most $\mathcal{O}(N)$. So in the primal algorithm, the dominant complexity is for two whiles which is $\mathcal{O}(N^2)$. For the dual algorithm, sorting the demands of N nodes has a complexity of $N \log N$. For each node, updating π^*_k and c^k_n requires at

most T minimization and T for new values. Each minimization has a complexity of T log T. Therefore, the total number of operations is at most N log $N + N(T$ log $T + T)$. Since $T \sim \mathcal{O}(\log N)$, the number of operations is no more that$N$ log $N(2 + \log(\log N))$. So, the complexity of the dual algorithm is $\mathcal{O}(N$ log N log(log $N))$ □

Proof of Theorem 5

Theorem 5. *For a solution $(\mathcal{X}, \mathcal{Y})$, let $f(\mathcal{X}, \mathcal{Y})$ be the objective function. If $(\mathcal{X}^*, \mathcal{Y}^*)$ is the optimal solution of $(\mathcal{SNCE}_{TWO})$ and $(\mathcal{X}^A, \mathcal{Y}^A)$ is the solution returned by the approximation algorithm, then,*

$$f(\mathcal{X}^A, \mathcal{Y}^A) - f(\mathcal{X}^*, \mathcal{Y}^*) \leq \sum_{(i,j)\in\mathcal{A}} \left(c^{up}_{1ij} + c^{us}_{1ij}\right) + \sum_{i\in\mathcal{N}} \left(c^{vp}_{1i} + c^{vs}_{1i}\right).$$

Proof. We know that $f(\mathcal{X}^{LP}, \mathcal{Y}^{LP}) \leq f(\mathcal{X}^*, \mathcal{Y}^*)$. Therefore,

$$\begin{aligned} GAP &\leq f(\mathcal{X}^A, \mathcal{Y}^A) - f(\mathcal{X}^{LP}, \mathcal{Y}^{LP}) \\ &= f(\mathcal{X}^A, \mathcal{Y}^A) - f(\mathcal{X}^{LP}, \mathcal{Y}^A) + f(\mathcal{X}^{LP}, \mathcal{Y}^A) - f(\mathcal{X}^{LP}, \mathcal{Y}^{LP}) \\ &\leq f(\mathcal{X}^{LP}, \mathcal{Y}^A) - f(\mathcal{X}^{LP}, \mathcal{Y}^{LP}), \end{aligned}$$

where the last inequality holds since $\mathcal{X}^A$ is an optimal capacity allocation corresponding to $\mathcal{Y}^A$, while $\mathcal{X}^{LP}$ is just a feasible solution. Therefore, $f(\mathcal{X}^A, \mathcal{Y}^A) \leq f(\mathcal{X}^{LP}, \mathcal{Y}^A)$. Now we have:

$$\begin{aligned} f(\mathcal{X}^{LP}, \mathcal{Y}^A) - f(\mathcal{X}^{LP}, \mathcal{Y}^{LP}) \;=\; & \\ & \sum_{(i,j)\in\mathcal{A}} \sum_{n\in\mathcal{T}} p_n \left[c^{up}_{nij}\left(u^{pA}_{nij} - u^{pLP}_{nij}\right) + c^{us}_{nij}\left(u^{sA}_{nij} - u^{sLP}_{nij}\right)\right] \\ & + \sum_{i\in\mathcal{N}} \sum_{n\in\mathcal{T}} p_n \left[c^{vp}_{ni}\left(v^{pA}_{ni} - v^{pLP}_{ni}\right) + c^{vs}_{ni}\left(v^{sA}_{ni} - v^{sLP}_{ni}\right)\right]. \end{aligned} \tag{26}$$

Note that we can analyze the sub-problems for nodes and arcs separately.

For any node $i \in \mathcal{N}$,

$$\begin{aligned}
&\sum_{n\in\mathcal{T}} p_n \left(c_{ni}^{vp} v_{ni}^{pA} + c_{ni}^{vs} v_{ni}^{sA}\right) \\
&\quad = \quad \text{Min} \quad \sum_{n\in\mathcal{T}} p_n \left(c_{ni}^{vp} v_{ni}^{p} + c_{ni}^{vs} v_{ni}^{s}\right) \\
&\qquad\quad \text{s.t.} \quad \sum_{m\in\bar{\mathcal{P}}(n)} v_{mi}^{p} + v_{ni}^{s} \geq \left\lceil \frac{\sum_{(i,j)\in\delta_i^+} x_{nij}^{LP}}{C_i^v} \right\rceil \quad \forall n \in \mathcal{T} \\
&\qquad\qquad\quad v_{ni}^{p}, v_{ni}^{s} \in \mathbb{R}^+ \qquad\qquad\qquad\qquad\quad \forall n \in \mathcal{T} \\
&\quad = \quad \text{Max} \quad \sum_{n\in\mathcal{T}} \left\lceil \frac{\sum_{(i,j)\in\delta_i^+} x_{nij}^{LP}}{C_i^v} \right\rceil \pi_{in} \\
&\qquad\quad \text{s.t.} \quad \textstyle\sum_{m\in\bar{\mathcal{T}}(n)} \pi_{im} \leq p_n c_{ni}^{vp} \qquad\qquad\quad \forall n \in \mathcal{T} \\
&\qquad\qquad\quad \pi_{in} \leq p_n c_{ni}^{vs} \qquad\qquad\qquad\qquad\quad \forall n \in \mathcal{T} \\
&\qquad\qquad\quad \pi_{in} \in \mathbb{R}^+ \qquad\qquad\qquad\qquad\quad \forall n \in \mathcal{T},
\end{aligned} \tag{27}$$

and

$$\begin{aligned}
&\sum_{n\in\mathcal{T}} p_n \left(c_{ni}^{vp} v_{ni}^{pLP} + c_{ni}^{vs} v_{ni}^{sLP}\right) \\
&\quad = \quad \text{Min} \quad \sum_{n\in\mathcal{T}} p_n \left(c_{ni}^{vp} v_{ni}^{p} + c_{ni}^{vs} v_{ni}^{s}\right) \\
&\qquad\quad \text{s.t.} \quad \sum_{m\in\bar{\mathcal{P}}(n)} v_{mi}^{p} + v_{ni}^{s} \geq \frac{\sum_{(i,j)\in\delta_i^+} x_{nij}^{LP}}{C_i^v} \quad \forall n \in \mathcal{T} \\
&\qquad\qquad\quad v_{ni}^{p}, v_{ni}^{s} \in \mathbb{R}^+ \qquad\qquad\qquad\qquad\quad \forall n \in \mathcal{T} \\
&\quad = \quad \text{Max} \quad \sum_{n\in\mathcal{T}} \frac{\sum_{(i,j)\in\delta_i^+} x_{nij}^{LP}}{C_i^v} \pi_{in} \\
&\qquad\quad \text{s.t.} \quad \textstyle\sum_{m\in\bar{\mathcal{T}}(n)} \pi_{im} \leq p_n c_{ni}^{vp} \qquad\qquad\quad \forall n \in \mathcal{T} \\
&\qquad\qquad\quad \pi_{in} \leq p_n c_{ni}^{vs} \qquad\qquad\qquad\qquad\quad \forall n \in \mathcal{T} \\
&\qquad\qquad\quad \pi_{in} \in \mathbb{R}^+ \qquad\qquad\qquad\qquad\quad \forall n \in \mathcal{T}.
\end{aligned} \tag{28}$$

Therefore,

$$
\begin{aligned}
&\sum_{n\in\mathcal{T}} p_n \left[c_{ni}^{vp}(v_{ni}^{pA} - v_{ni}^{pLP}) + c_{ni}^{vs}(v_{ni}^{sA} - v_{ni}^{sLP})\right] \\
&\quad\leq \quad \text{Max} \quad \sum_{n\in\mathcal{T}} \left(\left\lceil \frac{\sum_{(i,j)\in\delta_i^+} x_{nij}^{LP}}{C_i^v} \right\rceil - \frac{\sum_{(i,j)\in\delta_i^+} x_{nij}^{LP}}{C_i^v} \right) \pi_{in} \\
&\qquad\quad \text{s.t.} \quad \sum_{m\in\bar{\mathcal{T}}(n)} \pi_{im} \leq p_n c_{ni}^{vp} \qquad\qquad \forall n \in \mathcal{T}, \\
&\qquad\qquad\quad \pi_{in} \leq p_n c_{ni}^{vs} \qquad\qquad \forall n \in \mathcal{T} \\
&\qquad\qquad\quad v_{ni}^{p}, v_{ni}^{s} \in \mathbb{R}^+ \qquad\qquad \forall n \in \mathcal{T} \\
&\quad\leq \quad \text{Max} \quad \sum_{n\in\mathcal{T}} \pi_{in} \\
&\qquad\quad \text{s.t.} \quad \sum_{m\in\bar{\mathcal{T}}(n)} \pi_{im} \leq p_n c_{ni}^{vp} \qquad\qquad \forall n \in \mathcal{T} \\
&\qquad\qquad\quad \pi_{in} \leq p_n c_{ni}^{vs} \qquad\qquad \forall n \in \mathcal{T} \\
&\qquad\qquad\quad v_{ni}^{p}, v_{ni}^{s} \in \mathbb{R}^+ \qquad\qquad \forall n \in \mathcal{T} \\
&\quad= \quad \text{Min} \quad \sum_{n\in\mathcal{T}} p_n \left(c_{ni}^{vp} v_{ni}^{p} + c_{ni}^{vs} v_{ni}^{s}\right) \\
&\qquad\quad \text{s.t.} \quad \sum_{m\in\bar{\mathcal{P}}(n)} v_{mi}^{p} + v_{ni}^{s} \geq 1 \qquad\qquad \forall n \in \mathcal{T} \\
&\qquad\qquad\quad v_{ni}^{p}, v_{ni}^{s} \in \mathbb{R}^+ \qquad\qquad \forall n \in \mathcal{T} \\
&\quad\leq \quad c_{1i}^{vp} + c_{1i}^{vs},
\end{aligned}
\tag{29}
$$

where in (29), the first inequality comes from (27) and (28), the second inequality holds because $\left\lceil \frac{\sum_{(i,j)\in\delta_i^+} x_{nij}^{LP}}{C_i^v} \right\rceil - \frac{\sum_{(i,j)\in\delta_i^+} x_{nij}^{LP}}{C_i^v} \leq 1$, and the equality comes from duality. Finally, consider the case when the demand is 1 in all nodes of the scenario tree. Note that a feasible solution is to buy 1 unit of permanent capacity at the root node (which will satisfy the demands for all nodes except for the root node) and 1 unit of spot market capacity to satisfy the demand of the root node. This feasible solution will provide an upper bound $c_{1i}^{vp} + c_{1i}^{vs}$ for the optimal objective value. Therefore, the last inequality holds. We can repeat this reasoning for each arc and show that for all arc $(i, j) \in \mathcal{A}$:

$$
\sum_{n\in\mathcal{T}} p_n \left[c_{nij}^{up}(u_{nij}^{pA} - u_{nij}^{pLP}) + c_{nij}^{us}(u_{nij}^{sA} - s_{nij}^{sLP})\right] \leq c_{1ij}^{up} + c_{1ij}^{us}. \tag{30}
$$

The result will follow if we incorporate (30) and (29) into (26). □

The performance of LPA Algorithm for Large-scale Instances

We designed some large-instances of $\mathcal{SNCE}_{TWO}$ and solved them using the LPA algorithm. For these tests, we fixed B to 2 and tested the same network for different scenario tree sizes. The purpose of these tests is to show the efficiency and convergence of the LPA algorithm for solving large-scale instances that cannot be solved by CPLEX MIP solver. For these tests, we used a powerful server computer with an AMD Opteron 2.79GHz processor and 64.00 GB of RAM running Microsoft Windows Server 2008 R2 and we coded the tests in GAMS 24.2.2 calling CPLEX 12.6. The results are presented in Table 6. We should note that for these tests, since we do not have the optimal solution, the optimality gap is calculated as the relative difference between the solution returned by the LPA algorithm and the optimal objective value of the linear relaxation of the instances. This means that the actual gap between the optimal solution and the solution returned by the LPA algorithm is even smaller that what is reported in Table 6.

Table 6: LPA algorithm performance for large-scale instances of $\mathcal{SNCE}_{TWO}$

$\lvert\mathcal{N}\rvert$	Transient Nodes	$\lvert\mathcal{A}\rvert$	T	No. of Variables		No. of Constraints	Solution Time (s)	Optimality Gap (%)
				Real	Integer			
			3	6,860	14,420	7,560	28.02	4.64
			4	14,700	30,900	16,200	29.32	2.80
50	30	980	5	30,380	63,860	33,480	30.79	1.34
			6	61,740	129,780	68,040	33.20	0.12
			7	124,460	261,620	137,160	40.20	0.03
			3	27,720	56,840	29,120	116.75	3.71
			4	59,400	121,800	62,400	117.78	2.94
100	60	3,960	5	122,760	251,720	128,960	118.18	1.38
			6	249,480	511,560	262,080	125.66	0.53
			7	502,920	1,031,240	528,320	141.55	0.04
			3	174,300	352,100	177,800	705.81	6.58
			4	373,500	754,500	381,000	824.78	4.87
250	150	24,900	5	771,900	1,559,300	787,400	1030.28	3.37
			6	1,568,700	3,168,900	1,600,200	1871.54	1.13
			7	3,162,300	6,388,100	3,225,800	2841.56	0.04
			3	698,600	1,404,200	705,600	2803.02	5.47
			4	1,497,000	3,009,000	1,512,000	4319.53	4.82
500	300	99,800	5	3,093,800	6,218,600	3,124,800	5488.39	2.35
			6	6,287,400	12,637,800	6,350,400	7361.44	0.12
			7	12,674,600	25,476,200	12,801,600	21264.61	0.03